

AUGUST 1992 £1.95

2 4 6 RH COLUMN
7-10 11
13 14 25
32 34
38 ASA
40 43

SOFTWARE FILE

MicroEmacs

Qmenu



**FOUNTS OF
WISDOM**
text87
comes
into
the
90's



qqqqqqqqqqqqqqqqqq

Birthday Party!

10 YEARS OF THE SINCLAIR SPECTRUM



Editor
Helen Armstrong

Publisher
Mark Kasprowicz

Advertising Manager
Jim Peskett

Creative Director
John Stanley

Designer
Steve Billington

Magazine Services
Pauline Wakeling
Frances Maxwell
Linda Miller

Sinclair QL World,
Published By Arcwind Ltd.
The Coach House,
Tackley, Oxon.
OX5 3AH
Tel: 086 983 677
Fax: 086 983 733
ISSN 026806X

If you have any comments or difficulties please write to the Editor and we will do our best to deal with your problem in the magazine, though we cannot guarantee individual replies.
Back issues are available from the publisher price £2.00 UK, £2.75 Europe. Overseas rates on request.

Subscriptions from: Arcwind
Subscription Department, Lazahold Ltd,
PO Box 10, Roper Street, Pallion Ind.
Estate, Sunderland. SR4 4SN.
Tel: 091 510 2290
UK: £21.00
Europe: £32.00
Rest of World: £38.00
Please include your subscription number with any queries
Reprographic Services
Lithospeed, Scrutton Street, London.
Printed and Bound by William Gibbons,
Planetary road, Willenhall, West Midlands

© 1992 Arcwind Ltd,
Sinclair QL world is published
12 times a year. All rights
reserved. Reproduction in
whole or in part without writ-
ten permission is strictly pro-
hibited. We welcome
contributions but can not be
responsible for any unsolicited
contributions, all material
must be supplied with a SAE.

CONTENTS

AUGUST 1992

- 4** TROUBLESHOOTER ● Upgrades on the QL scene
- 11** QL SCENE ● Merz releases Q-Spread
- 13** QL SCENE ● Hermz IPC from TF
- 14** OPEN CHANNEL ● Assembler + C compiler
- 16** MEETINGS AND BIRTHDAYS ● The spectrum is 10 years old
- 18** INTO THE 90's ● text87 Plus4 comes of age
SOFT WARE FILE ● Cricket Secretary
- 24** CLUB ACCESS AND BINDER OFFER
- 26** THE NEW USER GUIDE ● INPUTS to LRUN
- 29** INSTANT ACCESS ● Instant addresses
- 30** BOOK REVIEW ● Another selection of old and new
- 32** SOFTWARE FILE ● MicroEmacs
- 34** DIY TOOL KIT Windows for Qdos, part 2
- 37** SUBSCRIPTIONS TO QL
- 40** SOFTWARE FILE ● Qmenu
- 43** SYSTEMATIC MACHINE CODE PROGRAMMING ●
Part 9
- 50** NEW QL's SPECIAL READER OFFER

**COMING
SOON!**

*We have the
new Hermz,
Intelligence
Periferal
Controller, and an
Infra-red controlled Ser-mouse
from Germany in the reviews box.
Plus belated reviews of QDesign
and Open World. Archive
Answers looks at bibliography*



T A R O U B L E

It is just as well that there are some contributors who use *Archive*, *Easel* and *Abacus*, as it is very easy for those of us who do a fair amount of writing to think only in terms of wordprocessors. I use a spreadsheet or database only occasionally, whereas it is a rare day when I don't use one or more wordprocessors. No doubt the statistics show that the majority of office and home micros excepting those designed solely for games-playing - are employed only for text work, like glorified typewriters. Nevertheless, integrated software packages always include modules for the other functions.

The trouble with *Archive* and, to a lesser extent, *Abacus*, is that they are harder for the average user to get to grips with than *Quill* is. The same can be said of similar programs. You may want only to create a name and address list, but are put off by the formalities you have to go through to get the job done. *FlashBack* made such jobs much easier, but it was effectively abandoned before it had been developed fully. Trying to explain to someone who has never used a computer before why it is necessary to have Fields, Records, Variables and so on is no easy task. In attempting to do this recently, I had to make it clear that what was being explained was the conventional view of constructing a database, not my own way of handling a list of addresses.

Using Ascii

Mainly because there seemed no other straightforward way of doing it, the transfer of my two *FlashBack* databases from QL to PC was done as a simple Ascii text file to a wordprocessor. No structure to bother about - just a bunch of fields within records, which ended up as individual paragraphs in the wordprocessor program. That was a year or two back, and the files still sit in the same wordprocessor program; there has been nothing to convince me that they ought to be re-created as 'proper' databases.

The only reason this approach can be successful is speed. The wordprocessor has to work fast enough to allow scanning through the databases at a pace which does not cause annoyance. My files are not large ones (about 60 and 120 KB). At the time the transfer was made, my PC was effectively much faster than the QL, but things have changed drastically since then. The *Gold Card* and *Perfection* have made the same exercise sensible on the QL. Digital Precision

In a world where upgrades are normal, we can always think of something else, says Bryan Davies.

keep records as *Perfection* files, not because they won't use programs from other suppliers but because it is the simplest and fastest way of handling the data. Nevertheless, some of us will still hanker for a program with the flexibility of wordprocessor but with part of the formal structure possessed by database programs. What this may mean is a combination of *FlashBack* features with those of *Perfection* or *text87*.

Taking a look at software for other computers, we can see database programs which allow data from the same file to be presented in several, visually-different forms, including those normally associated with spreadsheet and business graphics programs. One such that I have used for some years allows the screen to be split into four areas, one showing individual records in a top-to-bottom layout, as in *Archive*, another presenting groups of records in a linear, side-to-side fashion, a third displaying X and Y axes and several graphs plotted from the same data, and the fourth giving a boxed display somewhat like a spreadsheet, with cross-tabulated data. That last is a bit esoteric, maybe, although it can present a quite new perspective on sets of figures. Note that the discussion has gone from alpha to numeric data; there is no real reason why the same program cannot handle normal alphanumeric records, such as names and addresses, as well as doing calculations on the numeric data from the records. You can use *Abacus* for text preparation as well as for figure work.

Having got all the data, and the capability of manipulating it, why not permit reports to be printed out in a really presentable manner? 'Presentation Graphics' has an air of flash about it, but there are advantages to be gained from printed matter that uses some graphics (maybe only a logo).

We have had some very comprehensive and good developments in wordprocessor for the QL recently. What I am asking is 'can we now have some non-

wordprocessor functions added?' A little bit of *FlashBack*'s ease of handling records, some of *Professional Publisher*'s graphics-import capability, simple formulae as in *Abacus*, the typestyle handling of *text87*, and the general manipulative ability of *Perfection*. All rolled into one. Not much to ask, is it? Yes, we do need improved graphics to display it all, but the hardware stalwarts Up North are already working on that...

What is speed?

A recent article and advert may have led to some confusion, and amusement, among wordprocessor buffs. Is it really sensible to make speed comparisons between wordprocessor programs? Many people would want to see a straight 'no' answer to that question, but it really isn't that simple, is it? How accurate and valid can such comparisons be? *Quill* is so slow performing certain operations that any would-be competitor has to be able to show that it is very much faster. It wouldn't do to say a Copy operation takes half a day instead of a whole one - users want something better. With the current standards, any normal wordprocessor operation that takes minutes is too slow.

For instance, *The Editor* is quite a speedy program - until you use it in Document Mode, when it becomes much slower. Where possible, any documents used during comparison checking need to be typical of what significant numbers of users generate, and in the native format of the program under test. If quoting times taken to load *Quill* files into other programs, remember that this is generally an infrequent one-shot operation and, from there on, the imported file will be used in the format of the new program, so any speed checks should be made in that format.

We all have a threshold, below which improvements don't get noticed. My own, in connection with program functions, seems to be around 25%, but only when the time taken is appreciable. A block operation that takes 15 seconds instead of 20 is likely to seem faster to me. Whether or not one that takes 7.5 seconds as opposed to 10 will be noticed is another matter, and the chances of noticing 3 seconds rather than 4 are not good. At levels of less than 10 seconds, it is unlikely that most users will be impressed, unless an improvement is, say, 50% or more. We

M S O L V E D

[illegible]

less-and-less suitable as time goes by).

Still one up

Having had occasion recently to ask for a quotation on a high-end PC, I found with surprise that there do not appear to be any floppy disk controllers available which can support ED (2.88 MB) drives on PCs. Can it be that Miracle Systems are the only buyers of these drives in the UK? If so, all the more credit to them for bringing them to us, at a good price. It seems incredible that one can get hard disk drives with GB (gigabyte) capacity, CD-rom disks with 500-plus MB capacity, and 50 MHz processors, yet have to settle for 1.44 MB floppy drives as a maximum on the PC.

Readers' letters

Dilwyn Jones Software have written to reply in advance to possible criticism. As of early June, the update to the *Page Designer V 2 Plus* - had not been completed, and there is no firm date for the necessary work to be finished. Money that has been sent in is being held, and not banked, until copies of the program are ready to be shipped. The hope is that this will be by mid-July. The reasons given for the delay are pressure of other work and underestimation of the time needed for the program changes. (*All the usual!*) It is

refreshing to have a reply from a supplier before a complaint is received!

Digital Precision have asked that another warning be published about sending blank cheques through the mail. While DP is happy that customers place such trust in them, they are worried about the possible consequences of a letter containing a blank cheque going astray and getting in the wrong hands. It is quite possible for cheques to be cashed by people for whom they were not intended. If you must send a blank cheque to anybody, mark it 'A/C PAYEE ONLY'. This warning is made for users' own good; it must be stressed that nobody has complained to me about any cheque being cashed by the wrong person.

Brian Wray wrote to complain that Digital Precision had not refunded the purchase price of a copy of the Payroll program, although they had returned his master disk after checking it. DP replied that Wray's problem with the program had been caused by not observing the written instructions, and that there was nothing wrong with the program disk. The section of the program which deals with National Insurance contributions had to be updated by the user whenever the contribution rate was changed. However, they are contacting Mr Wray again

and will supply him with an updated version of *Payroll*; the delay was due to a problem over payment for the program. Other users who bought Payroll from DP (not from PDQL) should contact DP for details of the upgrade, if they require it.

PH Tanner wrote again - an even-longer letter than before. Long enough to fill this issue, in fact! Among many other comments, he said that "One of my machines runs 24 hours a day, all I/O taking place via NET". My own machines run 10-15 hours per day, but virtually never overnight, mainly because everything done on them requires my input, whereas Mr Tanner sets his QLs to calculating and lets them get on with it. Note the use of the network, and not with the aid of *Toolkit II* either. It is always heartening to hear of someone who has had a business lifetime working with better-known, and bigger, computers, but prefers to use the QL during retirement. In this case, the QLs certainly work for a living, too. Can anyone give an up-to-date report on the *GST Assembler*? In particular, has further de-bugging been done on it?

No reply has been received to the various complaints passed to **TK Computerware** in the past month or so.

A Quantum Leap in QL Wordprocessing

Our new state-of-the-art wordprocessor, **text87plus4** is now available in version 3. **MILES** ahead of any QL application!

text⁸⁷ plus4

+1 USER FRIENDLY TO THE EXTREME

- New well-written manual • Automatic setup and installation • Context-sensitive help • File selector boxes • Commands and key-presses highly compatible with Quill.

+2 THE MOST POWERFUL QL WP

- Extensive navigation and editing • Fast block and search / replace operations • Reformat as you edit • Remembers paragraph formats between sessions • Extensive file commands • Spell Checker with browse and replace • Multi-Window Multi-Document • Page-Preview and Pagination save a lot of time and effort.

+3 UNRIVALLED PRINT QUALITY

- Nothing else can compete in text and character formatting • Fully supports proportional spacing and justifies correctly • Multiple paragraph formats with different margins and line-spacing for each • Right, centre and decimal tabs • Multiple columns plus headers

and footers • Desktop publishing with several different page layouts in the same document

+4 FASTEST QL WORDPROCESSOR

Amazing timings on QL (much faster with Gold Card) and Atari ST with QL emulator. A 70 page text (24,000 words, 141,000 characters) was used • Load: 25s (STQL 17s) • Save: 37s (STQL 32s) • Search / replace (with auto-reformat) 580 instances: 43s (STQL 14s) • Change justification: less than 2s (STQL 1s) • Change right margin and reformat 65s (STQL 17s) • Move 10 pages from top to bottom (With manual marking and positions): 35s (STQL 15s) • Scroll screen 100 lines: 19s (STQL 5s)

Fully compatible with all QL ROMs, Gold Card, ST QL, Thor. Requires disk drive and 256K memory.

Prices (inclusive of Air Mail to overseas)

text87plus4	£ 79.00
2488 drivers for 24pin and Bubblejet printers	£ 19.00
typeset90-deskjet drivers for all HP Deskjets	£ 19.00
typeset90-Epson GQ5000, EPL4100 /7100 lasers	£ 39.00
fountext88 + founted89	£ 39.00

Graphic driver for 9 & 24-pin printers with over 30 founts

Ask for our free, comprehensive leaflets.

SPECIAL OFFER! If you have paid over £59 for any other QL wordprocessor you can upgrade to **text87plus4** for only £59. The manual from the other software is required as proof of purchase (It will be punched and returned). Offer expires end of August 1992. Send cheque, Eurocheque, to: **Software87, 33 Savernake Road, London NW3 2JU**

QL S C E N E

Fax rolls for printers

Reader P J Hudson has followed up his comments in Bryan Davies' column (page 4 of the April 1992 QL World) on the use of fax paper with the **Serial 8056** printer, by contacting supplier Office World (full name Globus Office World Plc) of Reading. Their Summer (May to August) catalogue ('We guarantee that our prices are absolutely rock bottom!') lists various sizes of fax roll paper, used by some QL World readers as a convenient source of printer-paper for reel-fed machines. The relevant price list can be found on page 107 of the catalogue under Fax Rolls.

Mr Hudson points out that the 30m rolls have the correctly-sized core for use with the Serial 8056, and his custom of re-winding the 100m rolls is simply to take advantage of economies of scale. Office World deliver.

Office World has stores in **Leicester (0533 516222)**, **London (081 961 8244)**, **Northampton (0604 34456)**, **Nottingham (0602 599499)**, **Reading (0734 568865)**, and **Slough (0753 551234)**. Enquire there for up-to-date prices and catalogues.

Clarification by Software 87

At the time of placing their advertisement in the June 1992 issue of Sinclair QL World, Software87 (the producer of *text87*) was unaware of the existence of the Digital Precision program *Perfection Special Edition*. The benchmarks quoted by Software87 in that advertisement hence definitely did not refer to the Special Edition of Perfection, which is understood to be much faster than the standard version of Perfection. In addition, Software87 would like to point out that the benchmark for Search and Replace quoted by Software87 in the same advertisement may have been unfair to the program actually referred to. Software apologises to Digital Precision for any inconvenience caused.

Speccy Emulators in Italy

Ergon Development is bringing to light two **Spectrum emulators** for the QL. The ZM-2 and ZM-3 ZX Spectrum 48K emulators emulate all standard Spectrum features (keyboard, tape reading and saving on Qdos devices) except sound. The two versions give a choice of facilities: the ZM-2 runs more slowly than the ZM-3, but has a higher compatibility rating. For instance, on a 16 MHz Gold Card games run on the ZM-2 from 35% to 70% of normal Spectrum speeds, while on the ZM-3 they run from 50% to 90% of normal speed. Speed, of course, depends on various factors, such as screen redrawing, screen mode, instruction type executed, and so on).

As for compatibility, some of the games quoted as running 'exceptionally well' on both emulators are Jetpac, Pssst, Sabre Wulf, The Hobbit (*kill Thorin*), Atic Atac, Manic Miner, UnderWorld and Knight Lore. The emulators are controlled by the 'Supervisor', which allows users to manage Spectrum files on QL devices, to disassemble/dump or poke the Spectrum memory, to redirect Spectrum keys to QL cursor keys, and more.

The complete ZM-X (ZM2 and 3) system costs £35 and is supplied with QL programs (on disk) and Spectrum programs (on tape) for transfer to the QL through the network or the serial port, for example, with the Interface 1). Ergon have also developed a full working public domain version, the ZM-1.

Prices quoted do not include carriage (£5 airmail postage). Please send cheques or (preferably) Eurocheques in Italian lire (£1 - 2200 lire), OR add £4 to cover conversion fees if sending any currency cheque other than lire.

Ergon also offer discounts on total orders: £3 off for two programs, £6 off the three programs, and so on. Please make payments to **Davide Santachiara**.

Orders and enquiries to **Ergon Developments, Via Emilio de Marchi n°, 42100 Reggio Emilia, Italia. Tel (Italy) (+39) 0522 70409**.

Ergon have also just released **Open World 2.1**, to allow QL users to convert graphic screens created with other computers into standard QL screens. It can import IFF (Amiga), CUT, TIF (PC and Mac) and GIF files (mainly from bulletin boards). Images can be converted into four colours, eight colours, or monochrome (with a very powerful dithering algorithm) screens. The monochromes are ready to be loaded into QL desktop publishing programs. Open World (OWR) costs £25 and comes with a supervisor program, and two 720 K disks full of GIF, IFF and TIF images.

Merz releases QSpread

Merz Software's **QSpread spreadsheet** for the QL, running under the QJump Pointer Environment, is now ready for market. The early news from Jochen is that the program is fully pointer- and keyboard-driven, can split windows three times vertically and horizontally, giving a total of nine working windows; is formula-oriented, so that formulas can always be retrieved, even after numerical input has taken place (unlike Abacus); allows function macros; allows a total of 32,767 cells, laid out in whatever horizontal and vertical configuration the user chooses. Block handling and number value entry has been made 'a lot easier' than in Abacus.

As usual, all necessary pointer environment and menus material is included in the package. Qspread will cost £49 from **Jochen Merz Software, im stillen Winkel 12, D-4100 Duisberg 11, Germany. Tel. 101 49 203 501 274**.

Further news from Merz: the **SerMouse** driver is now in version 2.00, with smoother mouse movement and other improvements. The QL Emulator for the Mega STE is in prototype, and is expected to be ready by August. More news when that arrives. **QDesign 2**, the upgrade to QDesign, improves on the original Qdesign in a number of ways, but the incompatibility with early QL rom versions (according to our report, this includes early Minervas) remains.

QL SCENE

EEC Drives a Bargain

EEC Ltd. are continuing to produce their improved MGT Lifetime Disk Drive, which is now sold as the **EEC Universal Disk Drive**. A 1 MB version is available at the reduced price of £70 (VAT-inclusive), and the 2 MB version at £90, which is half the original MGT price. The price includes an instruction book, a free 3.5 in disk, and a connecting lead for any one of the following computers: Sinclair Spectrum, QL, IBM PC or compatible, BBC, Amstrad, Atari, and Amiga computers. Extra leads are £12 each.

A kit of parts can be bought, consisting of a case, a transformer, DIP switches, and a data socket, for £23 plus £5 postage. This is for users wishing to make their own drives or customised units. EEC can also supply 1 MB and 2 MB drives at £29 and £49 respectively, and power supply units for 5V drives at £6.

The 1 MB version will format up to 720 K with

double-density disks. The 2 MB version (which also reads double density) will format up to 1.44 MB, providing the system will support it.

EEC are still supplying QL and Spectrum computers and hardware including microdrive expansion kits, microdrives and parts, as well as cartridges, although these are no longer manufactured. All prices include 17.5% VAT.

Bill Richardson is also stocking the Serial 3-Button Mouse at £45, and the Infra-red Remote Mouse at £55. These fit externally into Ser2 with no internal fitting required. They come complete with interface and software. According to advance publicity, the 3-Button mouse offers five more functions than the Qimi.

Orders and information from

W N Richardson & Co., 18-21 Misbourne House, Chiltern Hill, Chalfont St. Peter, Bucks SL9 9UE. Tel. 0753 888866.

Hermes steps in where the 8049 leaves off

Hermes is a replacement **Intelligent Peripheral Controller** (ipc) chip developed for the QL by Minerva's Laurence Reeves, and marketed by Minerva's regular agent, TF Services.

Hermes is designed to replace the QL's own ipc, the IPC 8049, which has presented users with certain problems since the QL was invented. Among the best-known problems with the original IPC 8049 are poor handshaking on serial input (sometimes requiring a full reset); keybounce, especially with early PC add-on keyboards; lack of independent baud rates; inaccurate sound functions.

These are among the problems which Hermes cures. The literature lists more precisely: two-key rollover, even with Shift; fully reliable serial input up to 19200 baud with one stop bit, assuming that the IC25 output buffer is working correctly and handshaking is enabled at both ends, and connected (check your serial cable and make sure the ports are open); differ-

ent baud rates for each ser1 and ser2 input lines, and separate from output (can be used to drive a serial mouse and a printer from ser1 simultaneously. Open/closed status of serial ports can be read; code to handle three spare I/O lines (one is used by the Qview capsled kit) Three other input lines can be read; keyclick toggled on and off; Reset/INT77 invoked more safely; function to return ipc version number.

Fitting involves removing the top of the QL and replacing the IPC chip. full instructions are included. Also included is a small extensions file to be loaded with RESPR.

The Hermes package costs £25 inclusive of chip, disk, VAT and UK postage, from **TF Services, 12 Bouverie Place, London W2 1RB. Tel. 071 724 9053.** Personally, we would wrap up our old chip and keep it in a safe place, rather than filing it where Laurence recommends - why contribute to a landfill when you can clutter your cupboards?

OPEN CHANNEL

Open Channel is where you have the opportunity to voice your opinions in *Sinclair QL World*. Whether you want to ask for help with a technical problem, provide somebody with the answer, or

just sound off about something which bothers you, write to:

Open Channel, Sinclair QL World, The Blue Barn, Tew Lane, Wootton, Woodstock Oxon. OX7 1HA

Assembler

Earlier this year you asked about an assembler and a C compiler for an unexpanded QL. The only assembler I have ever used on the QL is the Assembler Workbench, published originally by Talent. It is well adapted for the unexpanded QL. Not only that, but it comes with excellent on-line help, and a good range of monitor and debugging facilities. It also has a great little text editor with it, for writing programs. I use this to write Forth programs as well as assembler listings. It lacks some of the facilities of bigger and more expensive packages. The most important

factor is lack of macros. However, you only really need these if you are writing big programs which would require memory expansion, and so you would not use them on an unexpanded QL anyway. Assembler Workbench has never lacked in any facility I found any need for, and as it seems to be cheaper than any other assembler, it is almost certainly the best buy for someone who has an unexpanded QL.

As for C compilers, I have dabbled in C for a couple of years now, both on the QL and on IBM compatibles. But I have not written any serious program in it. It clearly is a very good language for those involved with large programming projects, if only because

of the enormous libraries of routines available to start you off. But for small programs (less than, say, 32K) it is much too complicated a process to go from source code to final program. It is so much quicker and easier to program in assembler, or Forth, or even Basic, and then compile the Basic.

To use almost any C compiler will require memory expansion, and even if it works with an unexpanded QL, it is unlikely to be very useful. However, if I were going to start experimenting with C on the QL, I think I would get the public domain C68 C compiler from Qubbesoft. It's good, it's continually getting better, and it's free (apart from copying charges). It's the only one I have actually tried on the QL, although I have tried several C compilers for IBM compatibles.

Alan Bridewell
Swindon

Editor's notebook

Summertime, and the living is rainy. But not as quiet as we normally expect in July-going-on-August. This is no doubt a side-effect of our many moves with letters arriving thick and fast, either wanting to know what is going on, or congratulating us, or both. PaulinE and Fran are correcting the Subscribers list by hand, which may mean more rational overseas addressing in the future.

The hardware and software community is also hard at work. In particular we have new software news from Jochen Merz in Germany and Ergon Developments in Italy. We also have (slightly delayed) news of the Hermes replacements IPC chip for the QL, produced by Minerva's Laurence Reeves, and marketed by TF Services.

I won't pretend that we've entirely caught up. We're behind with reviews and the open Channel proofs are emerging in a funny order - but we're getting there.

Racket

Are you a QL criminal? In you live in that little island north of France, and have a personal computer then, undoubtedly, you are. David Drysdale must have difficulty speaking with his tongue firmly in his cheek.

I bought my QL in March 1985, and used it for writing letters. The first thing I did was to produce a cartridge with names and addresses, so that I could print envelopes and merge addresses for business letters, without having to type them. I now have 600 addresses on three disks, duplicated, of course.

I can openly state the above, because in Andorra we do not have such an act as the Data Protection Act of 1984. For the last two months, doubtless in all innocence, there have been articles in Quanta specifically trying to teach users how to enter addresses and telephone numbers using Archive.

The greatly-increasing use of electronic diaries and notepads

means that people other than computer users are doing the same, with the addition of telephone numbers, what the person or company is or does, their partners' names, etc., etc. PC programs are commercially available for this purpose.

The act was made law for very good reasons, but at the time the use of personal computers had not taken off as it has since. The act is certainly still needed, but it must be amended so that you are not all criminals in future.

I think that computer magazines, computer manufacturers, and others concerned should write letters to the newspapers, the Government, lobby their MPs, and generally highlight the absurdity of the particular section of the act.

This is intended to start the ball rolling, instead of everyone behaving 'criminally' and doing nothing to get the situation changed.

Gil Lamb
La Massana
Andorra

E-mail

Some news that might interest **QL World** readers. Any QLer who has access to the worldwide network that links many mainframes should get in touch with Giuseppe Zanetti who is collecting the names of QL users. His e-mail address is 'beppe@alesia.dei.unipd.it'. He will supply a list of names and e-mail address on request. There are only about 30 so far, short of the 100 required to set up a news group devoted exclusively to the QL, so we have decided to post QL-related news items on 'comp.os.misc'. Professor Timo Salmi in Finland has also made his QL programs available on an anonymous FTP server, 'garbo.uwasa.fi', these can be downloaded over the net, and he will also accept files from others to add to the collection.

I haven't attempted to explain the details of how to access the net

Howard Clase
St John's
Canada

Thanking you in advance, I remain:

M L Huppe
Lauwerecht 201
515 GR Utrecht
The Netherlands

Comment: If anyone can compose the short answer to this broad and rather personal question, we will print it in Open Channel. Meanwhile, we will consider the possibility of publishing the long answer, but not in Open Channel.

Calloo, callay, oh frabjous day,
and all that. It is great to have you
all back in circulation. Long may
you continue so.

Only one thing clouds my view: The QL remains so exactly matched to my needs that I find it difficult to believe that anyone else could find a use for it. I live in terror that the bubble will burst and I will wake up one morning to find myself the lone surviving user.

But I am not writing only to congratulate; there are two other matters, less important but which should nevertheless be brought to your attention.

The font used for the listings in Open Channel is not wholly suitable. The minus signs are almost invisible; the distinction between colon and semicolon is very fine; and the zeros really ought to be slashed.

Which brings me to the second problem which concerns me. You printed a scrap of my own code

on page 15. Sadly there is a transliteration error. A double vertical line ($\|$), signifying a bitwise OR, has become a double underscore in line 1040. It may have been my fault: my printer prints a vertical line as a broken vertical line, and it is my usual habit to fill in the gap by hand. It is possible that I omitted to do so.

If anyone should need a listing straight from the source, I would be happy to supply it.

Funny to see your comments on the colour of the fields. In my pre-war East Anglia youth they were bright yellow too, but a much 'cleaner' colour: they were growing mustard for the Colman mills in Norwich.

Please keep going. Your efforts are greatly appreciated.

P H Tanner
Glasgow

Comment: Having had another look, I am going to vote a disagreement with you on the clarity of the listings print: I find the colons and semicolons are distinct, and the minus signs quite clear, if a bit small. The 0 and the O are among the most distinct I have seen in any typeface. However, this is the first comment on this subject I have had from a reader, and I would be very pleased to see further opinions.

I may have read the missing vertical lines as broken lines, but the underscore is a translation error - it was in order last time I looked at it. I think this is one of the few characters which is going to go on giving us trouble.

It's a bit sad that oilseed pollen upsets so many people, especially as it's a close cousin of mustard, and surely no more non-organic than hay (as in 'hay fever', of course). A friend from Norwich says: 'I'd appreciate it more if there wasn't so much of it.'

Having recently purchased a Sony camcorder (from Comet, with a subsequent refund of over £50, because of the price of a competitor) I thought it might be possible to do titling with the QL. I contacted Jonathan Oakely, who explained that as far as he knew, it was not possible for the QL to run a Genlock system (which is necessary if you want to merge two video signals).

I then tried using Page Designer 2 to make titles and photographs from the monitor, not very successfully due to reflections. With

```

100 WMON
110 WINDOW #0;512,256,0,0
120 CLS #0: REMark BLANK SCREEN
130 WINDOW #0;512,10,0,246:PAPER #0,0:INK #0,2
140 WINDOW #1;448,60,32,0:PAPER #1,0:INK #1,7: REMark INPUT WINDOW
150 WINDOW #2;448,186,32,60:PAPER #2,0:INK #2,7:REMark DISPLAY WINDOW
160 CSIZE #1,3,I
170 CSIZE #2,3,I
180 DEFine PROCEDURE RULE
190 PRINT #1,""/""""""""??"?""""""""""
200 END DEFine
210 DEFine PROCEDURE CLSALL
220 CLS #1:CLS #2:CLS #0
230 END DEFine
240 ERT HOT_REMV (C)
250 ERT HOT_CMD ('C','CLS #1'): REMark CLEARS THE INPUT WINDOW
260 ERT HOT_CMD ('I','input":prg$',',print #2,prg$')

```

Toolkit 2 users should lose line 140 and replace 250 and 260 with:

```
250 ALTKEY 'C','CLS #1','"':REMark CLEARS THE INPUT WINDOW
260 ALTKEY 'I','input'":prg$','print #2,prg$','"
```

further thought, it was obvious that a monochrome signal could be used as it was the required composite video signal, but experiments in linking the monitor lead to the 8mm video recorder via the edit unit not only gave the title, but also all the unwanted rubbish on the screen.

It became apparent that what was needed was a completely blank screen which was the full 512 by 256 pixels. The nearest I could achieve was to have a one-line (512 by 10 pixel) window 0. Paper 0, Ink 7, Csize 3,1: a nine-line (448 by the remaining 186 pixels) window 2. With no borders, this gives a blank screen and the red ink on black in window 0 makes any typed-in commands almost un-noticeable on the video screen. The 448 width of windows was arrived at by experiment in order to keep the characters away from the edge of the screen, which however led to another small problem, which was easily solved. A strip each side of the windows remained uncleared. To remove this, I decided to first define window 0 to the full 512 by 256 pixels, CLS #0 and then redefine the windows.

Input into window 1 is transferred to window 2, and then CLS #1 clears window 1 (Alt-C). Two procedures are used: Rule is a rule which appears in window 1 to facilitate spacing and CIsall clears all windows. Although limited to the QL's character set, this is a very cheap option for titling (character generators start at £150). If you think computer equipment is expensive, try video: camera £395 to £1999, video recorder £220 to £1799, and an editor with mixer and fader/en-

hancer facilities £59 to a full editor system including character generator with memory £4999. Of course, my equipment costs the lowest of these prices. I hope the listings will be of use to those who can only afford the older type of cameras with no built-in titling facility.

Peter Rowell
Quanta and Cambridge
Archimedes and 68 Group
(Case)

Referring to Peter Tomlin's letter (), Abacus 2.00 accepts the dollar sign \$. What this does is to make cell references absolute. For example, if cell A1 contains 10, cell B1 contains 20, and cell C1 contains $\$A1 + B1$, cell C1 will show on screen the value 30. Echo cell C1 (use F3:E:C1 across range C2:C4) and cells C2 to C6 will now contain the value 10. If you move the cursor onto cell C2 (for example), in the bottom left of the screen the formula $\$A1 + B2$ will appear - A1 is an absolute reference, whereas B1 was taken as a relative reference.

To use the dollar sign, type it in as part of the formula, for example: B1 + B1 * SC2/100, entered into cell D1, could be used to calculate a VAT-inclusive amount, if C2 contained the current rate of VAT.

Rich Mellor
Walsall
W Midlands

Lively in London

Dilwyn Jones reports from the recent Quanta workshop in London

The recent workshop on the 30th of May 1992 in a church hall in North Kensington was organised by the London group affiliated to Quanta, the UK QL User Group. The venue was easily accessible both by car and by public transport.

Quanta workshops are mini-shows where members (and often other users) can come along and meet other QL users, meet QL traders to buy items for their QLs, ask questions, get help with their problems and attend informal lectures and discussions organised on a wide variety of subjects.

Non-members can come along too - admission is usually free and there is no obligation to join Quanta, although you may want to do so when you see all the activity and benefits which can be obtained from membership of the group.

Most of the well-known QL traders came along to set up shop, including Digital Precision, Miracle Systems, DJC, EEC, Qubbesoft, QBits, Liberation Software, CGH Services, CL Systems and Software 87.

Freddy Vachha of Digital Precision gave an all-day demonstration of his company's software for the QL and talked at length on a number of subjects. Miracle Systems showed their range of hardware for the QL. The Gold Card is now apparently selling very well following the recent price reduction to £225. Miracle were displaying what I think was the smallest QL serial-to-parallel printer port converter I have ever seen - the circuitry was small enough to fit into the cover of the printer connector. The device I saw was only the prototype unit, but we believe it will be on sale soon.

Trump swap

Miracle may be launching a number of new QL products this

year, so we eagerly await them. Their well known Expanderam card and Trump Card will now be produced and sold by Qubbesoft - Ron Dunnett told me that the good news is that there will be a modest price reduction on these two items. See this month's *QL Scene* for more details.

Fred Toussi of Software87 demonstrated and sold the Plus4 version of the word processor *text87*. This is a remarkable program, well worthy of a demonstration the next time you see Fred at a QL show.

CL Systems exhibited their CQV1 QL Video Digitiser, now available in a version compatible with the Gold Card. Mr Lang sampled pictures onto disk for customers, using a video camera. TF Services were selling a new device called Hermes. This is a direct replacement chip for the 8049 IPC unit in the QL, providing a number of improvements. It helps to solve the serial input problems on the QL, giving split-input baud rates on both serial ports, and much improved keyboard de-bounce. It can provide three input/output lines and many more features. The price was £25.00. Contact TF Services for further details. Tony Firshman said that both the standard Minerva eprom and the MKII version are still selling well.

CGH Services' stand had a number of leisure programs for the QL and a selection of QL magazines such as the QL Technical Review, QL Leisure Review and the International QL Report, an informative magazine originating in the USA, but with subscribers all over the world.

Second users

Several traders were selling a number of second-user bits and pieces for the QL. QBits and Qubbesoft, for example, both sold various items of software, hardware, spares and old books and magazines - several visitors said it was

worth the trip just to browse through the contents of such stands.

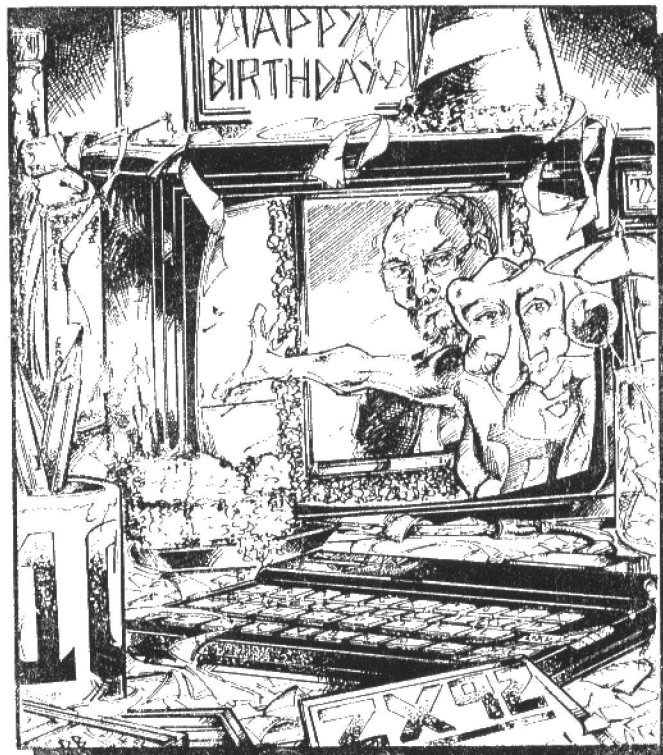
An interesting device was available from the QLEA stand. QLEA is the East Anglia Quanta sub-group. The QL ROM Switching Board is a small PCB which fits inside the QL and allows you to have two QL roms present, including a Minerva, which can be selected at the flick of a switch and a reset. A two-colour led indicator tells you which rom is in use. The benefit of this to users and software writers is that software testing is simplified - there is no need to dismantle your QL to change roms to test a program, or run a program which will not run with a Minerva installed, for example. The device is modestly priced at £25.00 plus a fitting charge (unless you fit it yourself).

Various talks took place on the stage area during the day, including Dave Walker's discussion of his excellent public domain C-68 C language compiler for the QL and Chris Boutal's talk on the fascinating subjects of Genealogy and his best-selling *QL Genealogist* program. Fred Toussi also gave a talk on *text87 Plus4*.

Due to the bad weather all over the country during the week before, this was not the best attended of Quanta workshops, although they are always great fun to attend. Quanta workshops are planned for Newcastle-upon-Tyne, Nottingham and Thetford later this year and possibly, by the time you read this, more venues will have been added. Why not go along and join in the fun?

The Spectrum's Birthday

Simon Goodwin held a birthday party for the Sinclair ZX Spectrum - 10 years old this May.



Britain's best-selling computer, the Sinclair ZX Spectrum, celebrated its tenth birthday this May, at a party organised by ZX enthusiasts in Cambridge, the Spectrum's birthplace.

The Spectrum followed Sinclair's ZX 80 and ZX 81 computers, and was known internally as the ZX 82. A decade on, after some seven million sales of Spectrum variants, ZX 92 was the obvious name for the anniversary re-union.

Spectrums, emulators, clones and eager users gathered at the Boat House pub, beside the river Cam. Star guests included John Mathieson, who joined Sinclair Research in 1981, wrote the original Spectrum brochure and tested the ZX rom before its release a decade ago. MGT founder Bruce Gordon brought old prototypes of his SAM micro, the true successor to the Spectrum, and new hardware under development.

Party music

The party featured a soundtrack of Spectrum-generated music, including Prokofiev's *March of the Capulets* rendered for 128K sound chip by muso Jon Bates, and *Overture for Toad*, an atmospheric MIDI sequence in the style of Philip Glass.

Cheeta's excellent Speedrum was used on many tracks, as were the RAM Music Machine and home-brewed software for effects and digital delays. Coventry band Relevant POS contributed catchy pop via their home-made interface for Wasp synthesisers, making a personal appearance with fans in tow.

Jamaican reggae and ska were represented by Norman Hall, plus guitar polemic from T1/T0, with former *Crash* and *Sinclair User* minion Garth Sumpter on vocals. Blues songs from Mel Croucher of Automata recalled the spirit of ZX Microfairs. Even Doctor Who made a contribution, in the shape of an out-take from Automata's classic Deus Ex Machina, narrated by Jon Pertwee!

John Mathieson brought two rare variants of the Spectrum: a white 48K model, produced to celebrate the millionth machine produced, and a unique Spectrum Plus, also moulded in white plastic as an experiment when Sinclair considered changing the case colour for the new model.

The original 'Issue One' Spectrum made an appearance, with the last-minute 'dead cockroach'

modification exposed - an extra chip soldered upside down beside the main logic array, needed to make early production keyboards work.

Downstairs in the bar another Spectrum development was earning its living. Delegates spotted the Treble Top pub game, a common arcade machine built around the Flare One circuit board. This super-Spectrum was invented in 1987 by three former Sinclair designers. It uses the same Z80B processor as the SAM Coupe, plus custom graphics and sound hardware.

The party attracted many luminaries from the Spectrum sub-culture, with a great spread of backgrounds and ages. The youngest delegate was no older than the computer, while the oldest had retired before the ZX range was born.

Fanzine writers included John Wase and Nev Young of *Format*, David Ledbury of *ZAT*, veteran adventurer Margot Porteus and Fractal guru Ettrick Thomson. Commercial competition was forgotten, as Glen Cook and Andy Wright discussed contrasting approaches in their SAM Game Design suites, due soon from Betasoft and Glenco.

ZX 92 was arranged by Sinclairphile and one-time *Crash* tipster Simon N Goodwin (me), and recorded for posterity by local media and Jon Pillar of *Your Sinclair*. Awards of software, publications and hardware were shared in a Spectrum-moderated draw. Top prize was a rare and valuable Currah Microspeech

voice synthesiser, donated by Betasoft.

Spectrum CD

Codemaster's Spectrum Compact Disk package attracted attention. This reads games from a normal CD player, linking the headphone socket to the Spectrum joystick port. You get 30 machine-code games, including several big hits, for £20 including the interface, ably demonstrating the size and economies of scale of the Spectrum market.

The range and quality of Spectrum software has encouraged micro users to develop 'emulation' packages that let them run ZX programs on other computers. William James' excellent Spectrum emulator for the Sinclair QL was on display, running ZX BASIC and popular games from QL disk. This emulator should reach the public domain once documentation and software to transfer ZX memory snapshots from disk have been finalised.

Already, Dave Barker's public-domain QSPEC software is capable of reading ZX cassette files and screens into memory, via the QL network port. QSPEC is available from CGH Services and other PD libraries. Spectrum file conversion programs are also available from Quanta (disk Comms Xfer 2) and CGH Services (Connections disk 1).

The Commodore Amiga has ZX emulators of its own, developed by Digimail of Milan, but the most technically impressive demon-

stration at ZX 92 was a double-emulation, with the QL Spectrum emulator running sweetly under the Public Domain Qdos emulator for the Amiga! The Qdos emulator was developed by Rainer Kowalik of Berlin; it lacks sound and supports only four colours, so some of the Spectrum hues appeared as stipples, but otherwise the double-emulation seemed perfect.

ZX on the PC

Andy Wright's ZX emulator for the Amstrad CPC range made a showing, alongside similar programs for the SAM Coupe, which was designed to run 48K Spectrum programs from the start. Bruce Gordon brought the original SAM prototype, hand-built on four circuit boards, with hundreds of TTL chips standing in for the custom ASIC in production models. He also displayed the latest model, shorn of MIDI and TV circuits to fit a paperback-sized board, and intended for process-control applications.

No less than three emulators for IBM PCs turned up at the party, via Germany, Spain and Holland, along with hardware emulators for minority British designs like the Memotech MTX and Tatung Einstein. It is still early days for emulation software, and you need a fast processor to match the speed of the original micro, but the future looks rosy as programs proliferate and chips rise to the challenge.



Text 87 Into the 90s

Bryan Davies at last investigates text87 Plus-4 in depth - delayed, but thoroughly stripped down and cleaned up. He likes it.

INFORMATION

Program: *Text87 Plus-4*

Price: £79

Supplier: Software87
33 Savernake Road
London NW3 2JU.

Speed is a good selling point and has been much emphasised in connection with wordprocessors. This review was written on the basis of experience running *text87 Plus-4* on a JS QL with a Mark 2 Gold Card and ED disk drives. There was also another 'tweak' in the system, in the form of the latest version of the *Lightning* display-enhancer program. These things smarten up the behaviour of almost any program, and it is easy to forget that life isn't quite so much 'in the fast lane' when your system doesn't have them. It is safe to say that lack of speed should not seriously trouble any user of Plus-4, though; it gets on with the job without painful pauses, in a way that *Quill* cannot manage.

The basic requirements for *text87 Plus4* are a QL (or Thor or Atari ST with QL emulator) having at least 256 KB of expansion memory in addition to the original 128 KB, one disk drive, and a display which can handle Mode4 text. The program version referred to here is labelled 'text87p4E3' ('T87' from here on), the final digit indicating release 3 of the Plus-4 program, current in early July 1992. This version has several enhancements, compared to the initial release version.

Not cosmetic

It is clear as soon as you start this program that it is no mere cosmetic face-lift of version 3. There is similarity in the menu structure and options, but a lot of changes have been made. That should not put off users of the existing versions, though, as they will find the style is essentially the same. There are simply more facilities available, and some of the existing ones have been made more usable.

There were problems with the initial releases (several different versions were received during the period of this review), but the complaints of users were heeded and the faults quickly rectified. No serious bugs were noticed in the final review copy. Files saved with very early copies have to be

loaded into the current version as Ascii text, and certain random characters at beginning and end of the text need to be deleted, to ensure compatibility.

The basic formula is still the same. A 'real life' screen, with dimensions in millimetres or inches rather than in terms of fixed-pitch characters, different display fonts to show both sizes and attributes such as italics, automatic reformatting when changes are made to text or to ruler parameters, ready-made printer-driver files to relieve the user of any fiddling, proper display of different justification modes and of proportionally-spaced characters, and on-line spelling checking. The new features include multiple windows, for 1 view of up to eight documents, or eight views of one document, or any combination. The windows are quite independent and spelling can be checked in all of them.

Getting started

Getting started is straightforward. A routine is provided to enable creation of a working copy of T87, setting the default devices, and selecting a printer-driver. The default Layout, Ruler and Typestyle settings can be altered by the user. Allowance is made in printer-drivers for non-printing areas on the paper; you need to bear this in mind when setting margins.

There is a fairly comprehensive on-screen Help feature, which should meet with general approval. The help text is initially loaded from disk, but then remains in memory (unless deliberately unloaded to release memory) and further use doesn't entail disk activity. F1 brings up the help, which is paged-through by means of the Shift-Up/Down cursor keyings. There are over 40 screens of help information, with eight lines on each, and the first screen you get is related to the commands menu you are currently using. Answers to most obvious questions should be obtainable from here.

T87 instruction booklets have met with complaint in the past and, in my experience, the best approach was to read all through the booklet - several times - before getting stuck into using the program. Some of the terms used were (and still are) unfamiliar, and the general style of the program was quite a bit different from *Quill* (which

was all most of us had met previously). The new booklet is well laid-out and clear to read. There is a Contents list at the front, but no Index (one is likely to be added, though); even in a relatively small publication, much time can be wasted looking for references to a particular word in the absence of an index.

Two menu structures

As a sign of understanding to those users who found the menu structure of previous versions too complex to work with, two menu modes have been provided. One causes all menu options to be displayed, in the appropriate groups, whereas the other deletes any options which are thought to be unwanted by users who are looking for only *Quill*-like commands. Thus, the File menu group - displayed when F3-F is pressed - shows Save / Load / Room / Close / New / Merge / Export / Import / Zap, in its full form, or Save / Load / Room / Close in its short-menu form. The mode is set as a configuration default which remains in force until reset, allowing the user to work with the short set-up for a few weeks, while getting the hang of using the program, then switch to the full menus as confidence builds up.

To a large extent, the special features of the program can be ignored; you can just start typing once the working copy has been made. To do this is a sad waste of useful features, but you still have the greater speed and reliability, and the availability of a spell-checking function, when compared to *Quill*.

Users of existing versions, and particularly those who have tried T87 and put it on one side, will ask whether or not functions are easier to use. Subjectively, my feeling is that there often tends to be one more keypress than is necessary to get the job done, but this is because of the effort that has been made to provide the maximum flexibility. The command names are mostly obvious ones, selection is generally by pressing the key for the first letter of the command, the Esc key takes you back one menu step (as it sensibly should do), you can restrict the number of menu choices and simplify matters by setting the 'short menus' default, the menu displays are clear and include some descriptive text.

For those who do not use a program every day - most of us? - the present structure is a sensible one, as you don't have to remember obscure key combinations; the 12-hours-per-day user will wish for some shortcut keyings, that can be memorised, and you can use the toolkit ALTKEY command for this.

Key combinations

Comparison with *Perfection* is inevitable, but generalisation on the relative merits of the menu systems is dangerous; what I will say is that - for me - T87 makes life easier by avoiding the necessity for remembering masses of key combinations, but more difficult by making access to commands less direct. How the individual user reacts depends to a fair extent upon his/her memory (or, indeed, the frequency with which the program is used).

One drawback to earlier versions was the inability to have more than one page layout in a document. You can now set up to 64 layouts per document; this ought to satisfy most users. All Layouts within one document must use the same page size and orientation, though, so you can't mix portrait and landscape pages. In addition, there can be several Frames per Layout, and there can be odd and even Layout pairs for facing pages. There can be different numbers of Columns in different Frames. The maximum number of col-

umns is 4; the gap between columns can be set.

The use of frames is similar to that in DTP programs, and will facilitate the creation of fancy newsletters and such. Text normally flows from frame to frame, as it does with linked DTP frames. Another new feature is a page preview. While this is far from being exactly what some PC users have got used to in recent years, it is comparable to that provided by Microsoft Word, for instance. The text is shown as solid blocks and is unreadable, whatever the size, but it allows you to get a reasonable feel for how the page will look when printed. The function is surprisingly fast. The lack of readability is presumably a result of the low resolution of the QL screen and should not be taken as a deficiency in the program.

Many windows

The availability of several windows makes life much easier, when your WP operations are less than straightforward. The first requirement was to be able to look at two different sections of the same, large document simultaneously, and that has now been fully satisfied; you can have as many as eight windows onto the same document. For anyone who tends to grass-hop (that definitely includes me when at the keyboard), an even more important requirement is to be able to have several different

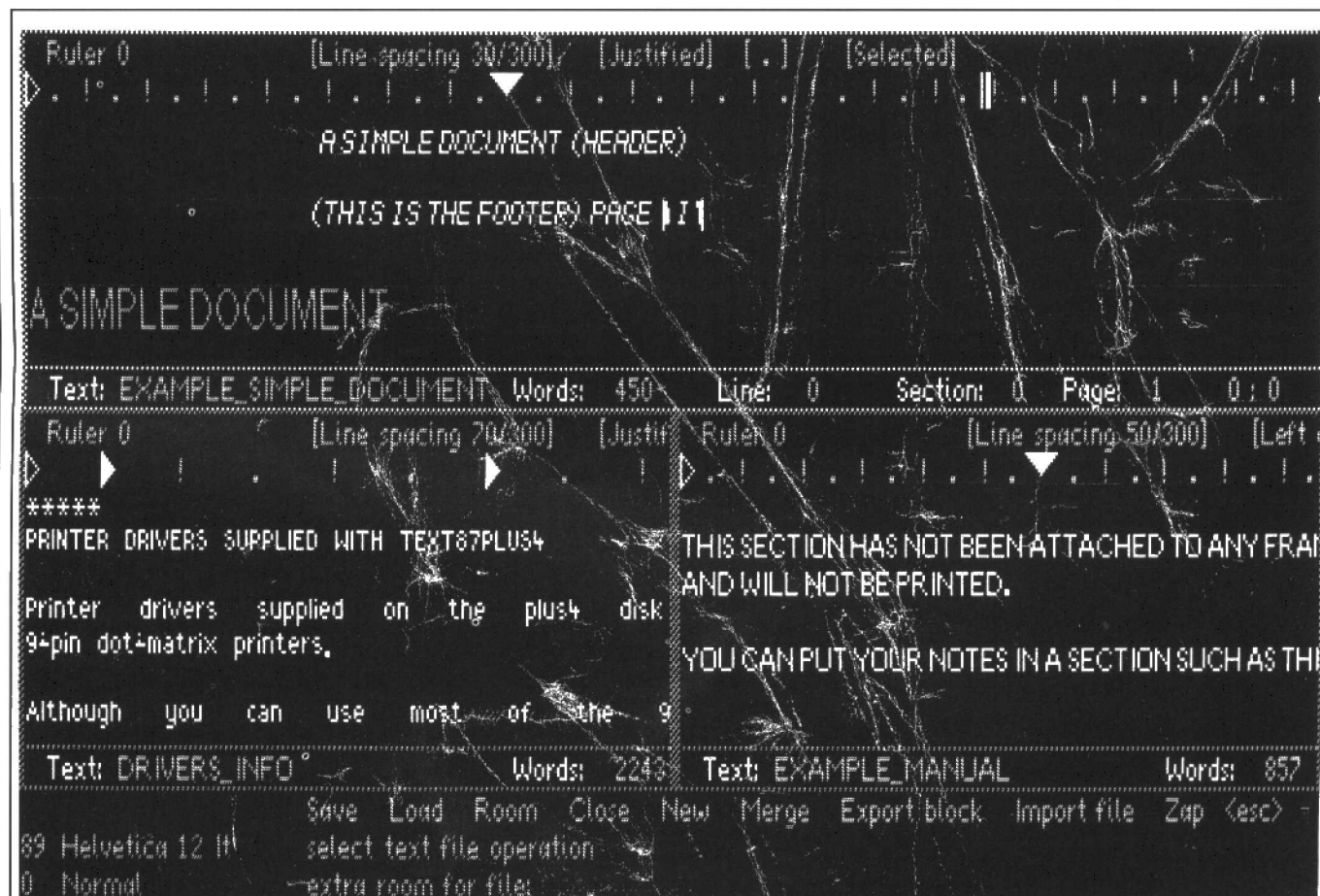
documents loaded at the same time, and be able to switch quickly between them for editing.

It is not an oddball need; for example, my current documents include this review, separate comments on T87 for the programmer, comments on a beta-test version of another program, and a letter to a computer nut in Germany. There was a further document containing replies to readers' letters a few days ago. All this is now possible, with T87. The windows can be sized and moved around. The ability to have a few words of many documents visible simultaneously does not cut much ice with me, as you get little enough text visible on a full 14 in screen as it is, but being able to switch quickly, and simply, between full-screen documents is very useful.

It is a pity that you cannot cut-and-paste directly from one document to another, but each document is unrelated to any others and you have to Export the desired block to (ram) disk, then Merge it into another document. But this does not apply to different windows onto the same document - you can mark a Block in one, then switch to another one and Move or Copy the Block.

Frame happy

The message 'margins too wide for frame' may be painfully familiar to many T87 users, but is presumably no more now, as the



program will carry on and print even if you have set the Ruler margins wider than the Frame width. You are warned that the results may not be too satisfactory, though. There is a change in terminology for the Ruler, as Indent is now First Line Margin. Normal indented or 'hanging' paragraphs are no problem; the First Line setting specifies the indent of the first character of the paragraph from the start of the Ruler, and the Paragraph Margin sets the indentation of all successive lines, again from the start of the Ruler. The Ruler is now panned to the left to accommodate markers when they are placed beyond A4 width, making it easier to see what you are doing with wide layouts.

We were all brought up on Quill, and came to expect to be able to look at a list of our document files, provided by Quill if the Load command was used. On 'graduation' to other programs, we found this facility absent, and we were forced to use Ctrl-C to go to SuperBasic to get a listing of files (displayed in a rather primitive fashion). It is good to see that Plus-4 has given us this important facility back again, in a very usable form. To obtain a listing of the files on the default drive; all you need do after selecting Load is press the up or down cursor key. The list contains only T87 and T91 (Plus-4) document files if you select Load, but all files are listed for other options. Moving the highlight to the one required, then pressing Enter twice, causes the selected file to be loaded. To check other devices, it is necessary to type in the device name (eg ram1_), then press the up/down cursor key.

Among the features which will be praised or cursed, depending upon the outlook of the user, are retention of the last cursor position when a document is reloaded. To my mind, this is a very desirable feature, as there are few things more annoying than always being taken to the start of a document after an editing session has been interrupted. When you Save a file, you are not asked whether or not you wish an existing file of the same name to be overwritten.

This certainly makes the save process faster, but can lead to heartburn. The current file name is displayed in the menu area when you request a Save, so you can't say you haven't been warned. It is not unusual to load a file of one name, modify it, then wish to save it under another name, and this is possible. In this case, you are asked if you really want to overwrite any existing file of the same name. The chosen name then becomes the default for future saves. The programmer can't win here, as it appears impossible to satisfy both types of user, without complicating life for all.

Block commands

Returning to the matter of speed, one area where Quill lets the user down badly is the copying or deleting of blocks of text.

T87 has no trouble with these operations, performing them quickly. The macro cursor movement command Go To has been made available within Block operations; although simply holding the cursor down gets you to another place quickly, it is helpful in long documents to be able to specify the page you wish to go to, when making a Copy or Move. The Search, and Replace, commands are performed similarly fast. Large documents do not appear to slow these commands down enough to bother about.

The spell-checking function utilises a version of the Q_Typ checker written by Tony Tebby. Although this is a separate function, the user is only aware of this initially, when the word list is loaded. Checking is called from the editing screen and behaves as an integral part of T87. The method of operation is that generally used on PCs, with the current document being checked as in a Search operation, each unknown word causing a halt and presenting the user with a list of options, one of which has to be chosen to set the process going again. The checker stops at anything which is not matched by an entry in its own word list.

The options are Ignore, Browse, Edit and Add To Word List. Choosing Ignore causes the checker to pass on to the next unknown word, without doing anything with the current one. Browse produces a list of words which begin with the same three characters as the unknown one; if one of those words happens to be what you actually need in place of the unknown one, you move the highlight (a red box, as in the other T87 functions) to the word and press Enter, whereupon the unknown is replaced by your selection. You are then taken back to the edit screen, and you can make further changes if required (such as if the replacement word needs an ending adding to it). Esc starts the checking process going again. Edit puts you in the same position as after Browse, to allow you to make your own changes to text.

Free movement

There doesn't appear to be any limitation on editing; you can move the cursor off the current line and go up or down the document, which can occasionally be desirable (maybe you've just noticed something which needs changing a few lines back).

'Add' does what would be expected, putting the current unknown word into a list which is supplementary to the main word list. The supplementary list as well as the main word list is checked for unknowns during the remainder of the current session, but the supplementary list is not saved automatically unfortunately. You must use the Save option of the F3-Config-Spelling menu to have your words recorded for future use; you must also use the Load option from the same menu - or the initial menu shown when spell-checking is selected - to have the supplementary list

available during the next checking session. This seems unwieldy, but is perhaps preferable to using the separate editor program to incorporate word lists into the main word list file, which is the recommended method of dealing with supplementary word lists. It was this latter operation which put me off using the spell-checker with the earlier version of T87, so things are better now.

There is no concurrent spell-checking function, but that shouldn't be missed by many users. As with many spell-checkers, upper case is inadequately dealt with but, even so, it was surprising to find 'germany' accepted as correct, once 'Germany' had been added. One should bear in mind that the spell-checker is included in what would be a very reasonable price for the WP program alone; the checking performance is good enough to justify a much higher price, especially now the larger of the two word lists supplied is said to contain over 200,000 words, and there are German and French lists as well as English, at no extra charge.

The end of the Page

The coming of hard disk, then the 3.2 MB floppy drive, brought with it the possibility of using sub-directories, a development for which most QL software was unprepared. Plus-4 supports the basic devices MDV, FLP, DEV, NET, RAM and WIN and interprets these letters followed by one digit and an underscore at the start of a file name as being a device name; that is, you can use DEV8_QUILL_ or such as a default storage device.

A seemingly small thing like the end of a page can cause a surprising amount of discussion. The earlier T87 didn't show the ends of pages unless you used the Go To Page command to force the display of red lines. Plus-4 is much more helpful, with three types of red line, for soft page breaks (where the program ends pages on the basis of the set page length), hard page breaks (where the user opts to end a page, regardless of length), and Section breaks; all are displayed automatically. The Section is a new concept for the QL, but is fundamental to the creation of many reference documents. Books are frequently divided into sections which may well have their individual page-numbering sequences. The simplest example is the first few pages - introduction, contents, acknowledgements, etc. They are not part of the main stream of the written text and should be numbered separately - i, ii, iii or similar. Page numbers can be printed automatically, anywhere within a frame; a marker code is used to tell the program the desired position for the numbers.

It is desirable to have all your 'working conditions' constant, and to have the current formats saved with a document. If the Ruler is displayed and the menu is at the bottom of the screen while you are editing

a document on one occasion, there is a fair chance that is how you will want the screen to appear the next time you load that document. More importantly, the positions of margins and tabs, the type of justification, text founts and attributes, and such should be preserved, along with the cursor position. The F3-Config-Save option allows you to set your defaults for having the Ruler display on or off, having the menu displayed at top or bottom of the screen, the amount of Room allocated to the next document, and the Ruler settings (margins and tabs), but you must use the F3-Config-Driver-Save option to preserve settings which are related to the printer, such as the default Typestyle.

View screens

In the gap between the introduction of versions 3 and Plus-4, much has happened in the PC world, and it is no longer sensible to say that T87 is far better at the business of WYSIWYG. The pseudo-operating system Windows on the PC has spawned a collection of wordprocessors which make a fair job of displaying founts in the correct sizes and shapes, and making it easy to select them (trouble-free these are not, however.) There is also good agreement between screen and paper appearance of text.

You may feel the need to do some ex-

perimenting, to get the on-screen founts to match those which are printed, reasonably well. When your printer is a laser or ink-jet type, the wide range of point sizes can be only approximately represented on the screen. Matching is much easier to do when you have a simple 9-pin DMP printer, with few founts. Apart from the screen appearance though, you don't have to worry about the effects of changing typestyle, because the program knows what you have requested, and takes care of the changes at print time.

The main area in which the screen does not mirror the printout, and can cause you to mess-up printouts, is in line spacing. There is an option which allows you to switch on or off representative spacing on the screen, but this does not affect what is sent to the printer; that is set manually, on the Ruler. If you switch from, say, 10-point to 18-point, you'd better change the Line Spacing setting to match, or you'll have lines printing over each other.

Running a test

To provide a general check on capability, a test document was created, with another program and on another type of computer. The size of the file was close to 300 KB, and it consisted of nearly 50 pages of text. There was no point in trying to retain formatting information, such as founts and

attributes, so the file was saved in 'Generic WP' format, which is similar to Ascii text (*a blessing on its little head - Ed.*). The file was then transferred to the QL by means of the XOver file-transfer program. Attempts to load it with earlier versions of Plus-4, using the File-Import-Ascii command were met with refusal, and no indication of why.

From previous experience, it looked likely the problem was lack of working memory space, and possible confirmation of this was given when the Merge command was tried, a 'short of memory' message being displayed. The old memory size command has been replaced by the Room command, which is not the same thing; it specifies the amount of extra editing space available for the next document, not the total memory for the document. However, the program was quickly modified to fix this problem and Import of such documents is now both simple and fast. The way the text appeared on the screen was a real pleasure - nothing funny about it, no non-printing codes, everything formatted as per the current Layout and Ruler settings, and it took only a few seconds.

Many users will want to load Quill files and/or old _T87 ones. The Quill _DOC file chosen for checking was an old standby, nearly 300 KB in size with over 40,000 words of text. It loaded with no bother, from the F3-File-Import-Quill command. The text attributes - bold, superscript, etc. - and tab

TEXT87 PLUS-4 NEWS

This page is A4 Landscape, split into 4 columns. You could use a similar layout to produce a newsletter in A5 size, with 2 pages of 2 columns each printed across A4 sheets. By giving a little careful thought to page numbering, and turning the paper over to print on both sides, you can create a booklet by folding & stapling the sheets down the middle.

The edit screen shows only 1 column; successive columns are displayed one below the other. The new Print Preview function *does* show the columns correctly, though; it also shows the justification of the text, which is not always apparent in the edit view.

There is a Gap function contained within the Columns option of Layout which allows you to put space at the middle of the page. As the gap has to be the same between every pair of columns, you can't set a decent space where the page will be folded without at the same time making the spacing between the pair of columns on each side rather

too large. It's a pity 5 columns aren't allowed, as that would enable the middle one to be left blank, but you could use 3 columns and settle for just 1 column on each A5 page, using the middle one for separating the pages.

There is another way of achieving the same result -- using Frames. The A4 Landscape Layout could be used, but divided into 4 thin vertical frames. As frames can be placed where you want them, the spacing between one pair can be different from that between another pair. This approach should produce the better result, at the expense of taking longer.

Even if you opt to use the Column option rather than Frame, you can still use the latter for any headers or footers you require. As the whole of the landscape A4 sheet of paper is 1 page, as far as Text87 is concerned, but 2 in the context of the newsletter, you may have to create, say, 4 frames to take separate headers and

footers for each of the 2 A5 pages on one side of a sheet of paper.

Don't hesitate to use the Print Preview, frequently. It is quick to produce a display of the "cameo" pages (PC programs tend to be very slow at displaying previews). Whilst you cannot read any of the text, you can certainly see whether or not the general form of it is going to look sensible when printed.

If you want to be a bit fancier, draw some lines to separate the columns. Text87 doesn't have a line-draw function -- particularly not in the vertical direction -- but what about using the underline style with the Space Bar and printing lines across the page in Portrait mode? This could be done before or after printing the text. For a regular publication, a master document with the necessary divider lines on it could be saved and re-used as required. A master could be set up for the column settings, too. Although Plus-4 lets you have several

Layouts in a document, you can't do that in this case, because the page size & orientation have to remain the same. You must reset the printer after the first run-through of the paper; it will have to be set to Portrait to print the lines, then Landscape to print the text, or vice versa.

The preceding point brings up another, in connection with founts. Text87 does not itself have both portrait and landscape founts and your printer may not have the same range in landscape orientation that it has in portrait. You might have to settle for a choice of just 2 founts in landscape mode.

A way round this particular problem is to use the *Fountext88* graphics printer-driver, which sends text to a printer as a graphics image, making it irrelevant what founts the printer has or doesn't have. The founts printed this way are by no means as smooth as those that are built-into printers such as the Hewlett-Packard DeskJet or LaserJet, though.

markers are preserved, but the actual tab positions change if the Tabs set on the current Ruler differ from those set in the _DOC file. The _T87 file selected was about three years old, and a mere 14 KB in size; this seemed less than a serious test, so four other _T87 files of roughly the same size each were appended to the first one, using the F3-File-Merge-Ascii command. Again, the earlier version gave a problem here, but that was promptly sorted out and all went well, and speedily, with the current version.

Ascii loading

The file format for _T87 and _T91 (the new type) files is not the same, and this is why the test files had to be treated as being Ascii. Formatting information and text attributes are preserved with files LOADED as _T87 files (but not with those subsequently MERGED as Ascii files). A SuperBasic boot file was loaded with F3-File-Import-Ascii also; the only things to note here are that the default Ruler setting is for full justification, which is unlikely to be suitable for program files (or indeed for many users' text files) and the margins need to be set wide enough for the maximum line length.

Some users will be interested in loading Perfection files. The two files that were checked loaded without hesitation, and the text format was basically correct. It was necessary to increase the Room (editing space) value to load a large file. The main market for T87, as for Perfection, is those users who are still clinging to Quill. You may well have to make layout, ruler and fount changes before, and possibly after, loading any 'alien' file, and there is no great problem in doing so, but it may take a fair amount of time, as it does with many other programs. There are Normal and Paragraph options for loading non-T87 files - the equivalent of Quill's By Line and By Paragraph options.

Memory-management is something no program seems to get fully right, from the user's point of view, but the Qdos operating system is not exactly helpful to the application programmer in this respect. Many users will be aware of the way Quill solves the problem by grabbing all the memory it can find. You know where you stand with Quill! T87 has a more difficult task; apart from the fact that your documents may vary greatly in size, you may wish to load several documents during the same session, and there is no way the program can anticipate the total memory requirement for all you wish to do. The Room command allows you to set the working space in memory, to some extent. As my first actions while doing this review came up against shortage of memory, it would have been nice to find an index in the instructions, with 'Memory' or 'Room' listed.

All in memory

The on-screen help tells you that the Room you allocate is expansion space for the next file to be loaded, and will be added to that taken by any files already loaded. The current program version has additional information on the screen concerning this function, and - as mentioned above - the commands which are connected to the Room function now work appreciably better. The instructions are to be modified also, to give more guidance to the user, and to incorporate an index. One thing to bear in mind is that the whole of a file is in memory, rather than part of it being on disk as can happen with long Quill documents, and this means faster operation.

It has been a plus-point of T87 for some time that certain printers can be used in their native modes, rather than being treated as simple, Epson-compatible types. The illustrations for my articles are printed on an Epson GQ-5000 laser printer. While using even only basic Courier 10-pitch on a laser printer produces a big improvement in perceived print quality over most dot-matrix printout, the difference becomes much more marked when you can use other founts and sizes which may be built into the laser. In the case of the GQ, the supplied printer driver (not part of the basic T87 package) uses the scalable Times and Helvetica founts, albeit only at certain sizes. The range of sizes is good, from 6 to 72 points, in steps of 2 to up to 14-point. If your printer supports printing in landscape mode, Plus-4 allows you to do this. The Layout must have suitably-wide margins; even with standard one-inch margins, you can print a spreadsheet with 116 12-pitch characters across landscape A4 paper. As with the extended Layout function, this brings you closer to having DTP capability.

Talking printers

Getting your particular printer to co-operate with a program has always been a major problem for users, and T87 attempts to ease this by giving a choice of 32 printer drivers in the basic package, plus other special ones available for 24-pin DMP printers, the HP DeskJet (covering LaserJet II emulation also), and Epson lasers, at extra cost. It is likely that most users will be able to find a suitable driver from those offered, and there is no need to get involved in any configuration procedure. Indeed, no provision is now made for any custom-tailoring, but anyone with special requirements can contact Software87 and ask if these can be met (don't assume changes will be done for free, though).

A feature that has gone missing is the visible fount/size code. You still have an indicator to show you what the current

selection is, but the codes are no longer seen in the text. Printing codes in general are something which can easily arouse strong arguments; my own preference is to have as many codes as possible either always visible within the text or, preferably, displayable only when the user wants them, but the average user is more likely to want them hidden all the time.

Criticisms? They were relatively minor with the earlier versions and have almost all been dealt with by the E3 version. It has never been of interest to me to know what line within a document the cursor is on, but this is what is shown on the status line, initially; to satisfy this complaint, an option has been added to set the default line numbering to within-page or within-document. The lack of automatic saving and reloading of the supplementary word list for the spelling checker is an irritation, but not a great one. The handling of memory might confuse the user, but it may be partly caused by the unhelpful way Qdos manages memory, and the latest program version certainly makes it much less likely most users will need to change the Room value.

Layouts and frames

The setting-up of Layouts and Frames cannot be called intuitive operations; you really need to read the instructions several times and look at the samples provided. Maybe the process cannot be simplified much, but it took me longer than it should have to get the hang of it. It would be useful to have some more flexibility in certain commands; on the other hand, a few less keypresses would be appreciated. There is, as yet, no advance on the previous graphics-import function (which was anyway provided by an add-on program), so that full DTP functionality is still some way off; the addition of the necessary features is a large job, and would justify a considerable price increase, whilst being of interest to only a limited section of the QL community. The comments made above about the on-screen representation of typestyles and line spacing should be viewed in comparison with what other QL programs offer; as has been the case from the start, T87 has to be put into the DTP category in this one respect.

Plus-4 is a big step forward from previous versions. You still have to work at using some functions, but the menu structure has been improved and more help is given now, both on-screen and in the instructions. The flexibility of document layout is much-increased, and the print preview takes the program into a more professional area. It is fast, with no operational problems apparent. Made for serious scribes who care what the printed output looks like. Good value for money.

SOFTWARE FILE

Cricket Secretary

INFORMATION

Program: Cricket Secretary.
A suite of programs to enable the Secretary of a cricket club to maintain the batting and bowling averages of his club.
Author: C B Storey.
Supplier: Dilwyn Jones
Computing
41 Bros Emrys
Tal y Bont
Bangor
Gwynedd
Price: £12.00
Mdv, 3.5 in or 5.25 in disks.
Works on an unexpanded Q.L.

Howzat! Here is a program especially designed for cricket lovers. To those devotees of the most English of games, watching is only half of the pleasure. The other half comes from writing down the scores, working out the batting Averages and then discussing them at length over a pint in the Local.

C B Storey has now developed the perfect program that does all the hard work for you. This, of course, leaves much more time for the discussion afterwards. Here we have a suite of programs, suitable for use by a cricket Secretary or enthusiast alike. All the statistics are available once the data has been entered:

Batting Averages
Bowling Averages
Best Bowling
Dismissal Averages

The program is supplied with an eight-page Manual that leads you through every aspect of the inputting and data retrieval system. I didn't find that it flowed as well as some I have read, and I recommend that buyers read it through thoroughly a few times before entering lots of data. It is not totally error-trapped, and a couple of times I crashed the program before I could save it on to a disk. This was due to a little ambiguity in the instructions.

The instructions start by telling you how to create the two disks required for the program to work. In fact, three are needed, one being a blank formatted disk to put the compulsory backup on. If you

Wotzat? It's John Shaw - trying to keep his speed down to the pace of a game of cricket!

try and get away without one, the first crash occurs. Once you have followed the set-up instructions, you are left with a master disk containing all the files needed to run the main part of the program, and a Data disk that will hold your scores and averages.

On running the program, you are asked which drives your master and data disks are on. Unfortunately, you have to do this every time you run the program. I found this a little tedious, and would have preferred to have had this configured permanently.

Then the Suite Menu appears:

1. Run Cricket Secretary
2. Run Printa
3. Run Toolkit
4. Quit

You make your choice by pressing the appropriate number. You will then be asked for the suffix of the files you wish to process. On the first occasion this is _0. This uses a skeleton program that forms the basis of future files. When you come to save them, you add a suitable suffix such as _som92 (Somerset 1992). You will eventually build up data files on the disk for every team or series you are collating.

That done, you are confronted by a menu offering the following choices: Input Batting, Input Bowling, Examine Batting Averages, Examine Bowling Averages, Examine Best Bowling, Examine Dismissal Analysis; Load new file/backup/return to menu.

Input Batting

On selecting this, you are first asked for the name of the cricketer. It is important that this is done in the correct format, that is: Shaw JR. (Neither SHAW JR nor JR Shaw. You have been warned!)

You then enter the number of runs scored. This is followed by an alphabetical input representing how the player was dismissed. A table of entries is provided to assist with this.

This sequence then repeats itself until you have entered all the batter's details.

Input Bowling

The principles are the same as for the Batting input. The number of Overs bowled is the first data to be entered. This is followed by the number of Maidens bowled, the number of Runs and finally the number of Wickets taken. The sequence then repeats itself.

Batting Averages

The batting data is computed to two decimal places and displayed in order of Average, the highest first.

Bowling Averages

These are computed to the same standard as the batting averages.

Best Bowling

This is a sub-set of the Bowling Average table, and comprises the Name, Runs per over, Balls per wicket and best Bowling.

Dismissal Analysis

This will show a table displaying how many times each batter has been dismissed and the method of dismissal.

Load New File

This is the compulsory Backup point. If you try to ignore the warning, you will crash and your data will be lost. Make sure you have a spare blank disk ready.

Printa enables you to print out hard copies of the data you have just compiled. You are given the option of installing a new printer driver if you wish. The Toolkit allows you to modify the database should you accidentally put in wrong detail. The author warns that you should take great care when using this facility. I agree!

This is a good and extensive program that, however, needs to be handled a little carefully in order to avoid problems. I offer you these few tips:

Have three blank disks ready at the beginning.

Read the Manual a few times before you start using the program.

Practice with a small amount of input in order to become familiar with the program before you enter all your scores.

Remember the way it requires you to enter the names, ie Brown GM.

Always keep a straight bat!

CRICKET SEC SUITE MENU

Copyright © C B Storey, 1992

1. RUN CRICKET SECRETARY
2. RUN PRINTA
3. RUN TOOLKIT
4. QUIT

*** Press the number you require ***

BOOK REVIEWS

Cornucopia of Algorithms

Title: *Practical Astronomy With Your Calculator*
Author: Peter Duffett-Smith
Price: £11.95
Title: *Astronomy With Your Personal Computer*
Author: Peter Duffett-Smith
Price: £15.95
Publisher: Cambridge University Press
Available: Currently in print
QL Specific: No

P *Practical astronomy with your calculator* is a cornucopia of algorithms for solving the often nightmarish calculations involved in astronomical mathematics. Do not be misled by the title: included is everything from relatively straightforward data and time calculations and conversions between co-ordinate systems to calculating the effects of atmospheric refraction, and the positions of the sun, moon and planets with a degree of accuracy which will satisfy all but the most exacting requirements. Times of the rising and setting of sun and moon, for example, can be found to within about a minute.

The mathematics used is not only eminently suitable for a personal computer but, for repeated calculations, makes one almost essential. Each section has not only a clear explanation of what is being done but at least one worked example.

The related volume *Astronomy with your personal computer* covers for the most part the same ground, though algorithms are given in a language

not unlike microsoft basic instead of step-by-step calculation. This translates easily to the QL's SuperBasic, the more so as the superbasic functions RAD, DEG, ASIN and ACOS can replace the more roundabout methods which Duffett-Smith uses in order to maximise the compatibility between dialects. In addition his modular approach lends well to the building of a library of superbasic procedures and functions which can be merged as appropriate for particular applications.

A number of the algorithms – particularly those which calculate the positions of the sun, moon and planets – differ substantially from the companion volume in being more accurate still, albeit at the inevitable expense of more typing-in and less speed in execution. (Finding the instantaneous position of the moon, for example requires 99 SIN and 32 COS calculations, even before the results have been worked into a useful format.)

One intriguing extra section provides a method of deriving a comet's orbital elements from visual observations: these can then be used to predict its future path.

A problem with SuperBasic which both of these books highlight is that of the numerical accuracy of constants. It will often be necessary to use something like $e = '23.441893'$ (with quotes) as otherwise SuperBasic will store any eighth and ninth digits internally only to lose them again if the line is subsequently edited or the program saved and reloaded. Entering constants as strings (which the system coerces to floating-point numbers when they are allocated) en-

sures that all significant figures stay where they are put, which makes a worthwhile difference with calculations such as these.

Having said that, even this reviewer's humble JS machine produced results which generally were closer to published predictions than the examples provided by the author, which were evidently run on an IBM micro. There is, however, much scope here for putting to practical use the accelerated maths routines of the *Minerva* rom or the *Lightning* maths extensions.

Not only astronomers but anyone with an interest in simply playing with mathematics will find enough in these splendid books to keep them occupied for hours.

Hilary Snaden

A Valuable Reference Manual

Title: *QL Advanced User Guide*
Author: Adrian Dickens
Publisher: Adder Publishing
Price: £12.95
Available: Fairs, second-hand
QL Specific: Yes

be gained over multitasking and similar internal functions of the machine. Each function is described, along with the methods used to invoke it, responses that may be given and parameters needed to be sent.

This list is preceded by a short introduction to QL machine code, an overview of Qdos and a listing of an experimenter which allows Qdos functions to be tried from Basic. For those new to machine code the introduction is, to my mind at least, too short to be easily understood. I feel therefore that the book cannot claim to assume no knowledge of assembler – but for those who know at least the basics it is excellent.

To demonstrate the uses of Qdos, the overview includes two assembler listings – one for a clock that displays the time permanently on screen and another for a set of rapidly reproducing, multitasking blobs that rush across the screen. Later listings show how to add procedures to Basic in machine code, how to control the QL's under-used two screen facility and provide a device driver for a parallel printer port.

Following the list of Qdos functions, the QL device drivers are described, as well as internal formats for most information used by the operating system. Finally, the machine code interface to Basic is examined.

The QL Advanced User Guide is described, on its back cover as 'the authoritative guide to the Sinclair QL System'. The description finishes with the statement 'No programmer of the QL can consider not owning a copy of this invaluable reference manual'. I could find little in the book to cause me to disagree with these statements – it contains a great deal of very useful information. However the back cover also claims that the book 'assumes no previous knowledge of operating systems or 68000 assembler' – something I would hesitate to accept.

The Guide is made up largely of a comprehensive list of the functions that the QL's inbuilt software, Qdos, provides for writers of machine code programs. These functions cover nearly any action that the QL may be seen doing, from saving data to printing to screen. By accessing them directly in machine code, full control can

This thorough description of the QL's internal workings is complemented by twenty appendices that cover additional details of the machine, from its memory map to hardware expansion information. These are very useful in their own right – even including a circuit diagram for a printer interface to match the device driver documented earlier. The book is completed by a bibliography, a glossary and full index. Certainly no QL programmer ought to be without this guide.

Andrew Toone

Integration of Machine Code and Electronics

Title: *Computer Interfacing – Connection to the Real World*
Author: Martin Cripps
Publisher: Edward Arnold, Hodder and Stoughton Ltd
Price: £12.95
Available: Currently in print
QL Specific: No

Martin Cripps, as the reader of this book will be quick to find out, is not satisfied. He is not satisfied with the current state of computer interfacing. As a Director of the Wolfson Microprocessor Unit and a Senior Lecturer at Imperial College, he has greater qualification to comment on these matters than most. In *Computer Interfacing* he sets out to explain his reasons for being unimpressed with the way in which we link our computers to their environment and, more importantly, ways in which those links can be improved.

Computer Interfacing was written to support courses presented to final year undergraduate students in computing, and as such contains some fairly complex explanations involving for instance, integration, machine code and electronics.

These are used as tools to describe through the book the way in which information is taken from external stimuli into the computer and the corresponding path for data from the computer to influence the real world.

This description is well defined as stages in the flow of information. Sensors are discussed, then

the sampling of data so as to accurately record an input waveform. Signal conditioning is explained before methods of digital to analogue and analogue to digital conversion; following these, the book goes on to cover the digital links to and between computers before standard interfaces are discussed. The pathway from real world to computer and back again is completed by a section on output transducers.

After these sections, Cripps rounds his subject by covering environmental constraints such as noise and power supply; microprocessor input-output; integrated interfaces (interfaces on a single chip); microprocessor architecture and design techniques, finishing with three design examples. The design examples perhaps best indicate the area in which *Computer Interfacing* is most useful. A gas-plasma display, automated greenhouse and a mobile robot are described in terms of the design decisions that would be taken to implement them. By stepping through the stages of interfacing the real world to microprocessors, the book covers all of the details involved in implementing any small microprocessor based system, from sophisticated engine management system to multiprocessing computer.

The book is well thought out and written in an informal style that lightens this otherwise complex subject. For anyone interested in computer hardware therefore, at a little under thirteen pounds it is a good value read – however, being written for fairly advanced students, some knowledge of the subjects involved is vital.

Nigel Bates

Drive Further than than Corner

Title: *Developing Applications on the Sinclair QL – Practical Ideas for Home and Business Use*
Author: Mike Grace
Publisher: Sunshine Books (Scot Press), 1984
Price: £6.95 in 1985
Available: At fairs or second-hand
QL Specific: Yes

If you've never believed that using a computer was meant to be fun as well as productive then Mike Grace's infectious enthusiasm for the QL and the bundled software might just change your mind.

Grace stresses that it's no good having powerful software unless you understand how that software works, and more importantly, how to apply it. Otherwise, it's rather like using a Ferrari to drive around to the corner-shop for a pint of milk.

Having said that, I must take the author to task for his misleading title. The book is concerned only with the Psion software and makes no reference to SuperBasic. Also, to judge by the examples he provides for each of the Psion packages, there is a very wide spectrum of meaning to his use of the word 'application'.

Grace rightly points out that the strength of the Psion software is that you can use it straight away without the tedium of learning to program. He lists a whole range of possible uses for students, home-users, club secretaries, part-time businesses and self-employed people – in short, anyone with a QL, although he does caution that some things are better done on paper than on computer. The emphatic message is: QL computing is for you!

The book is divided into five sections covering a general introduction to the QL and Psion software and each of the bundled software packages Quill, Abacus, Archive, and Easel. Grace relies on the reader to cross-reference with the User Guide – the main weakness of his book – and he

does not attempt a comprehensive description of all the commands. The longest section deals with Quill where the author encourages flair and creativity rather than viewing it merely as a kind of electronic typewriter. His account is comprehensive enough to enable the reader to use it effectively. However, the text needs to be read with the improvements of Version 2.35 in mind.

The section of Abacus is thorough and stimulating with interesting comments on the User Guide examples together with several of his own applications. There is a useful table of the most commonly-encountered Abacus commands and functions, and an illustration of the use of several of the more esoteric ones such as ave() and if(). However, Grace's diary/schedule-planner application is a good example of his own injunction that some things are better done on paper!

The section of Archive covers most of the commands and functions, illustrated by some simple database applications. His advice on the opening and closing of Archive files is misleading as database files are always at risk while they are open (as opposed to being looked at) even if they have been closed at some previous time. Grace does a particularly effective bit of soap-box shouting at the end of this section on how really good a computer the QL is. Unfortunately, the book starts to thin out a good deal from this point on.

Grace, like most other people, raves over Easel and what he calls its incredible flexibility, but you'll need to study the User Guide in much more detail than for the other Psion packages to get the full benefit of this.

At the end of the book, there is a glossary of both general and QL-specific jargon terms followed by a summary of chapter contents. Overall, there are few niggles with a book which is clearly laid out and written in a fresh and unpatronising style. For anyone who has not dared to think big with their Psion software or finds the User Guide too daunting, then I would recommend getting your hands on it.

Nigel Bates

SOFTWARE FILE

MicroEmacs

INFORMATION

Program: MicroEmacs text editor, recompiled.

Supplier: QubbeSoft and CGH public domain libraries

Price and format: See individual suppliers

Hilary Snaden tries the long-running MicroEmacs text editor, as updated and recompiled with the C68 C compiler.

Slow screen

MicroEmacs must be one of the longest-running sagas in the history of computing. Its origins lie in the pre-home computing days of the 1970s in the corridors of the Massachusetts Institute of Technology (MIT), where it was designed as a rough-and-ready text editor which could without too much difficulty be ported between different machines.

To this end it was written in C, the idea being that any machine with the basic hardware requirements which was capable of compiling the source code should also be capable of running the program.

The earliest versions were fairly basic, designed as a somewhat rudimentary aid to programming in C, but subsequent authors, most notably Daniel M Lawrence, together with a host of other programmers who have contributed bits of code along the way, have made MicroEmacs into one of the most sophisticated text editors available anywhere.

Most of the work in getting the program to work on the QL was done by Ken Norrie, who released a compilation with Lattice C of version 3.9p of the source code. More latterly, a number of minor bugs have been ironed out by Richard Kettlewell, and the code recompiled using the later and considerably improved C68 compiler. The result is a considerable increase in speed, though screen handling, the QL's perpetual Achilles' heel, is still somewhat slower than that of programs more painstakingly customised for the QL. So flexible is MicroEmacs, however, that it is possible to speed up the screen by altering the vertical and horizontal and vertical scrolling rates - in effect, by configuring a 'lazy screen'.

For example:

```
set $hscroll FALSE
```

will ensure that only the cursor line is panned rather than the whole screen, while

```
set $hjump 8
```

forces it to pan eight characters at a time. The difference these make is dramatic.

However, while the problematic colour modes now work after a fashion, changing from the default values seems to have the unfortunate effect of considerably slowing screen handling. However, the startup configuration of white text on a black background with a green mode line will probably be acceptable to most users, and it may well be that this minor bug will be fixed in future releases.

An advantage which MicroEmacs shares with the best QL software is that it multitasks from Basic with neither problems nor any need for fiddly extensions to SuperBasic. Indeed, since like Digital Precision's *Editor* it has no inbuilt directory facilities, multitasking via EXEC or EX is essential to make available SuperBasic's DIR command. The program code is fairly lengthy, but at some 98K it is only marginally longer than *Editor*, but with more to offer as a text editor; however, it should perhaps be stressed that MicroEmacs is not a wordprocessor. Most importantly, it has no facilities for driving printers.

It does have a number of functions rarely found in the most sophisticated of wordprocessors, and it is a relatively straightforward matter to insert control codes from chr\$(0) to chr\$(26) from the keyboard, so with some work the more common printer control strings can be embedded in the text.

Besides some standard wordprocessor-type functions such as fill-paragraph and set-fill-column (analogous to setting the right margin) there are some surprising extras. For example Ctrl-T swaps the character under the cursor with its nextdoor neighbour (*If only more wordprocessors had that facility*); presumably this was added by a programmer whose fingers, like those of many keyboard users, have mischievous minds of their own (or one who can never quite remember the rules about Is and Es).

It is also possible to convert spaces to tabs-and-spaces and vice versa; this can lead to a surprising saving in disk space with files containing data tables, or with a good deal of indenting such as C programs, and MicroEmacs can encrypt files so that your program or text

file is inaccessible without the 'password' used as an encryption key.

Word count

There are inbuilt functions for converting words or blocks of text to upper or lower case, and included in the count-words function is a display of average word length, useful as a check on writers inclined to verbosity or for those uncertain whether to send their latest work to the *Financial Times* or the *Sun*. This review, for example, averages 5.81 characters per word. Readers of *QL World* can make of that what they will.

An aspect of using the program which is a little confusing to begin with but which soon becomes a revelation is that Enter does not end a search or replace a prompt but adds <NL> to it. (The prompts are in fact terminated with ESCape.) This means that strings at the end or the beginning of a line can be specifically searched for, and double blank lines can easily be replaced by single. This is so useful it is surprising that authors of other editors have not tried to come up with something similar.

Any recognised combination of keys can be redefined to execute any of MicroEmacs' existing commands; hence, if you find yourself using some of the more obscure commands relatively often, you can bind them to a more convenient keystroke. The combinations available are Ctrl-(character), ESC+(character) and ESC+Ctrl-(character), leaving plenty of room for customisations. You can even redefine single keys, though this obviously needs to be done with some care!

In addition to the standard key bindings common to all versions of MicroEmacs (so, for example, should you have access to the same program on a Vax mainframe, Ctrl-N will go to next-line just as you are used to), the more usual cursor movements have been bound to the QL's cursor keys in such a way that they behave much as they do in *Quill* and the like.

Any deletions larger than single characters can be undone, since the deleted text is kept in a 'kill buffer' until the next major deletion. This is also how blocks of text are moved around.

Particularly astonishing is the speed of the replace-string command; presumably this is

largely because the screen is not updated each time a substitution occurs unless query-replace-string has been called.

By name

While all of the most commonly-used commands are already bound to short key sequences, all of those available can also be invoked by name - though one has first to know that ESC-X brings up the 'execute-named-command' command! Even this can be made to look something like Quill and most other QL text-handling software by binding the command to the F3 key with the line

```
bind-to-key execute-named-command FN0.
```

This and any other program configurations can also be saved in a file called `emacs_rc`, which will then be read and set up when the program is first started. As a practical example of this, Qdos traps Ctrl-C before it ever reaches MicroEmacs, and so the count-words command will not initially be available from simple keystrokes. It can be made so by the line:

```
bind-to-key count-words M-H
```

which uses the otherwise unused combination ESC-H.

If you can't remember the name of the command you want, MicroEmacs can produce a complete or selective list, or typing a space at the prompt will fill in the name up to the next ambiguity.

Almost every aspect of the workings of MicroEmacs can be altered from within the program; there is, for example, an automatic Save facility which initially resaves the file being edited every 256 characters, but this count can be set by the user. A perhaps inconvenient but quite understandable exception is program dataspace, which must be set either at compile-time or with a command line such as:

```
EX flpl_emacs;"%122880"
```

which will set the dataspace to 120K. Note that you need QJump's *Toolkit 2*, or some other enhanced EXEC command to do this, otherwise the default of 64K will be used. The current version of MicroEmacs will, therefore, just fit in a 256K machine. MicroEmacs will work with less dataspace, but at a noticeably lower speed. As with other programs, though, if little headroom in terms of ram is left, disk access will be slowed as Qdos will have little space available in which to set up slave blocks.

Windows

One of the most interesting of MicroEmacs' features is that it is possible to work on several different files concurrently, and to divide the screen horizontally into a number of windows which may display different files or different parts of the same file. However, if the same portion of text is displayed in two windows, screen handling can no longer keep up with with even an average typing speed, though it seems to catch up eventually.

The buffers used for each file can have the main editing modes (for example, overwrite, automatic word-wrap and case-dependent searching) set independently of the others. Moving text from one buffer to another is quite straightforward.

Potentially the most powerful of MicroEmacs' facilities is the macro language. This works something like Editor's command files but in a far more sophisticated and flexible fashion: not only are the inbuilt text manipulation procedures available but there are a number of mathematical and logical functions and structures, so that a complete macro may end up looking like an Archive program. The 'programs' can be invoked from disk or a text buffer, or can be built into MicroEmacs as a numbered macro or named procedure. As a relatively unsophisticated example of what can be done, the listing shows a procedure which (after a fashion) rennumbers a SuperBasic program file.

Would-be programmers should beware of the pitfall that MicroEmacs does not recognise the abort-command keying (Ctrl-G) when it is in the middle of a while construct, unless it has been specifically programmed in, as in the example. This should really have been mentioned in the documentation, as it is so easy, even with a macro debug facility, to end up with a perpetual loop, particularly while developing applications.

Source code

The public domain disks include the complete source code (about 475K of it) and a further 182K of documentation comprising a users' manual, a learn-as-you-go tutorial, and a file which, while originally intended to provide on-screen help, may be usefully printed out as a crib sheet.

If you are familiar with C and have a suitable compiler (the C68 compiler with which the current ready-to-run version of MicroEmacs was prepared is also available in the public domain) you can even hack around the original code to produce your own customised edition apart from the configuration that can be done from within the program itself. Since the source is likely to remain under development for the foreseeable future, and it is in the public domain, there is little chance of users being left with abandoned software.

Given the amount of man- and woman-hours which have gone into programming MicroEmacs, and the range of facilities it has to offer, it must represent one of the best value programs ever.

MicroEmacs is available from CGH Services and QubbeSoft public domain libraries. At the time of writing the new C68-compiled version had not percolated through to the Quanta members' library. For those who wish to experiment further the C68 compiler may be obtained from the same sources.

Example MicroEmacs procedure

```
store-procedure rn
; line-number a region
narrow-to-region
end-of-file
set %lines $curline
beginning-of-file
set %newnum @"Start number: "
set %inc @"Increment: "
write-message "Renumbering..."
!while &les $curline %lines
    !if Spending
        set %a &asc &gtk
        !if &equ %a 7
            write-message "[Aborted]"
            widen-from-region
            !return
        !endif
    !endif
beginning-of-line
search-forward " "
delete-previous-word
insert-string %newnum
insert-space
set %newnum &add %newnum %inc
next-line
!endwhile
widen-from-region
write-message "Renumbered!"
!endm
```


DIY TOOLKIT

Part two

Simon Goodwin wraps up his QL windowing project, begun in last month's QL World.

In the last issue, *DIY Toolkit* introduced seven new keywords which allow QL programs to save, compress and restore overlapping windows. This article presents the remainder of the assembler source for these keywords, explains their inner workings, and discusses further applications of the code.

If you missed the last issue, or do not wish to re-type my listings, you can obtain complete source and object code for Volume W, *DIY Windows*, on disk or microdrive cartridge, with documentation and extra example programs. Write to *DIY Toolkit* at Cwm Gwen Hall, Pencader, Dyfed, Cymru SA39 9HA, or call 0559 384574.

Send a stamped self-addressed envelope for details of the volumes currently available. *DIY Toolkit* Volumes cost £3 each, plus £4 per order to cover disks and processing costs. If you have no disk drive please enclose one cartridge for each volume required.

Icons for Qdos

The window routines can be used to display and highlight small pictures, or 'icons', on the QL screen. These may be stored on disk and displayed with a short SuperBasic loader. You can design these with programs like *GraphicQL*, *The Painter* or *Eye-Q*, and extract them from a screen image file with *W_STORE* or *W_CRUNCH*.

It is convenient to use positions and widths that are an exact multiple of eight pixels, such as 16, 24 or 32 pixels. Monochrome 32 by 32 pixel icons occupy 1/128th of the screen, and use 136 bytes each; four colour 16 by 16 pixel icons weigh in at 72 bytes.

You may prefer to use monochrome icons 16 pixels square, taking 40 bytes each, and *SET_RED* 1 to highlight one, changing a green on black image to white on red. These file sizes exclude the ALCHP heap overhead of 20-27 bytes per allocation.

With *DIY* icons and window control, you are well on the way to a complete 'WIMP' system of Windows, Icons, Menus and Pointers. The missing bits can simply be hacked together in SuperBasic, but you may prefer to use ready-made components, such as *Qmenu* from Jochen Merz. Tony Tebby's pointer interface provides a standard way to read mice, joysticks or the keyboard; it appears on Quanta library disk Specials 0.

```
* SET_RED #ch% , flag% [I=Set W=Clear -I=Copy to red bytes]
*
setred      bsr.s      get_channel    Read channel number
            subq.w     #2,d3          Check parameter count
            bne.s      bad_param2     Two are expected
            move.w     2(a1,a6.l),d6   Fetch the flag
            bsr.s      find_window    d6
            tst.w      d6
            lmi.s      red_lines      Copy green bytes to red
            addq.l     #1,a4          Point at odd (red) bytes
            beq.s      fill_lines     If D6=0, leave it alone
            moveq      #1,d6          Otherwise, set all bits
            lea.l      fill_bytes,a3   Put D6 in alternate bytes
            bra.s      scan_lines
red_lines    lea.l      red_bytes,a3   Point at next line
            bra.s      scan_lines

* bad_param      moveq      #-15,d0      BAD PARAMETER error
            bra.s      quick_exit
chan_error    moveq      #-6,d0         CHANNEL NOT OPEN error
quick_exit    addq.l     #4,a7          Tidy the return stack
            rts                    Return subroutine errors

* SET_GREEN #ch% , flag% [I=Set @=Clear -I=Copy to green bytes]
*
setgreen     bsr.s      get_channel    Read channel number
            subq.w     #2,d3          Check parameter count
            bne.s      bad_param2     Only one is expected
            move.w     2(a1,a6.l),d6
            bsr.s      find_window    d6
            tst.w      d6
            bpl.s      check_pat      Set or clear even bytes
            addq.l     #1,a4          Start with the red byte
            lea.l      green_bytes,a3 Point at byte copier
            bra.s      scan_lines

* FIND_WINDOW - locates window details from channel definition
*
find_window   lea.l      window_ext,a2 Point A2 at EXTOP code
            moveq      #-1,d5          Wait indefinitely long
            moveq      #9,d0          SD.EXTOP trap key
            trap       #3             Call via Qdos
            tst.l      d0              2 signals ERR,OK
            bne.s      quick_exit     Return error directly
            move.w     d1,d0           D0 is window line count
            move.l     d1,d4           Preserve width & height
            movea.l    a1,a4           Preserve base address
            swap       d1             Recover line width
            mulu       d0,d1           Compute storage needs
            tst.w      d6             Are we crunching?
            beq.s      no_crunch
            lsr.l      #1,d1           Halve space needed
            addq.l     #6,d1           Allow for size & depth
            addq.l     #6,d1           Allow for link & flag
            move.l     d1,d5           Preserve size needed
            swap       d4
            move.w     d4,d0           Retrieve line width
            lsr.w      #1,d0           Count words not bytes
            subq.w     #1,d0           Adjust for DBRA later
            swap       d4
            subq.w     #1,d4           Adjust for DBRA
            rts

* bad_range      moveq      #-4,d0      OUT OF RANGE error code
            rts
bad_param2     moveq      #-15,d0      BAD PARAMETER error code
            rts
exit2          rts

* W_SHOW #ch% , base_address          Re-displays stored windows
*
show          lea.l      16(a3),a4      Point A4 past parameter 2
            cmpa.l     a4,a5           Are there 2 parameters?
            bne.s      bad_param2     Forget the first for now
            subq.l     #8,a5
            bsr        get_channel     Retrieve parameter 2
            movea.l    a5,a3
            addq.l     #8,a5
            movea.w     #118,w,a2      CA.GTLIN, get a long word
            lsr        (a2)
```

Code commentary

The first part of the source code for DIY Windows appeared in the previous *DIY Toolkit* column. The remainder is listed this month, with notes on the overall design. I shall start by discussing the first part, so you may like to refer to Listing one from last month's *QL World*.

The window routines have many common features: they all take a window as a parameter and scan through the corresponding area of QL display memory. It follows that much code is shared between the keywords, and you need to understand the whole in order to make sense of the parts. After the usual snippet to introduce the new commands and functions to SuperBasic, the code begins by fetching parameters for `W_CRUNCH`, the function which stores a window in compressed form.

Much of the code for `W_CRUNCH` is shared by the next command, `W_STORE`. The distinguishing feature is the value in register `D6`, which records the way the image is to be compressed. `W_STORE` sets `D6` to zero, indicating that both red and green bytes are to be stored. `W_CRUNCH` copies its second parameter to `D6`; this is a QL colour-code: two for red or four to store only green bytes.

Both functions use the subroutine `GET_CHANNEL` to fetch parameters. This evaluates all the parameters as integers, using the `CA.GTINT` vector, which leaves the number of values found in register `D3`. `W_STORE` checks for a single parameter, the channel number. `W_CRUNCH` also expects the compression indicator, so it reports a 'bad parameter' error if `D3` is not set to two after the call to `GET_CHANNEL`.

Both these extensions return a value, so it is important not to leave unwanted values on the maths stack, where they might get in the way of the result. `GET_CHANNEL` tidies the stack with `ADDQ.L #2,$58(A6)` after reading the Basic channel number, and returns with the corresponding Qdos channel ID in `A0`. `W_CRUNCH` uses the same instruction to unstack the compression indicator.

`GET_CHANNEL` checks for several potential errors and returns directly to the SuperBasic interpreter with an error code in `D0` if a problem is found. The code at `QUICK_EXIT` discards the subroutine return address so there is no need to check `D0` after each call to `GET_CHANNEL`. They only return if all went well.

Overflow report

`GET_CHANNEL` reports 'overflow' or 'error in expression' if the parameter is not a valid integer, and 'bad parameter' if it is negative. It signals 'channel not open' if the channel has been closed or never opened.

`W_STORE` and `W_CRUNCH` share code from `SAVE_WINDOW` onwards. First

```

bne.s      exit2
move.l     @a1,a6.l,d7
bmi.s      bad_param2
btst       #0,d7
bne.s      bad_param2

* Now D7 is the buffer base address and A0 is the channel ID
*
movea.l    d7,a2
move.w     10(a2),d6
bsr.s      find_window
movea.l    d7,a0
addq.l     #4,a0
cmp.l      (a0)+,d5
bne.s      bad_range
cmp.w      (a0)+,d4
bne.s      bad_range
addq.l     #2,a0
subq.w     #2,d6
bmi.s      word_lines
bne.s      byte_lines
addq.l     #1,a4
byte_lines lea.l     load_bytes,a3
word_lines lea.l     load_words,a3
bra        scan_lines

* Find SCR_BASE (A1.L) LINE_WIDTH% (D1.H) LINE_COUNT% (D1.W)
*
window_ext moveq     #40,d0
cmp.l      (a0),d0
beq         bad_param2
move.w     24(a0),d0
lsl.w      #2,d0
bclr       #0,d0

*
movea.l     50(a0),a1
adda.w     d0,a1
move.w     26(a0),d1
mulu      #128,d1
adda.l     d1,a1

*
moveq      #6,d1
add.w      24(a0),d1
add.w      28(a0),d1
lsl.w      #2,d1
bclr       #0,d1
sub.w      d0,d1

*
swap       d1
move.w     30(a0),d1
moveq      #0,d0
rts

* Line scanning routines, used by JSR (A3) in SCAN_LINES
*
save_bytes move.b     (a4)+(a0)+
addq.l     #1,a4
dbra       d2,save_bytes
rts

*
fill_bytes move.b     d6,(a4)
addq.l     #2,a4
dbra       d2,fill_bytes
rts

*
green_bytes move.b     (a4),-(a4)
addq.l     #3,a4
dbra       d2,green_bytes
rts

*
red_bytes  move.b     (a4)+(a4)+
dbra       d2,red_bytes
rts

*
load_bytes move.b     (a0)+(a4)+
addq.l     #1,a4
dbra       d2,load_bytes
rts

*
load_words move.w     (a0)+(a4)+
dbra       d2,load_words
rts

*
swap_words move.b     (a4)+,d1
move.b     (a4),-(a4)
move.b     d1,(a4)+
dbra       d2,swap_words
rts

*
save_words move.w     (a4)+(a0)+
dbra       d2,save_words
rts

```

they call FIND_WINDOW to read details into registers from the channel definition of the window. This in turn calls an SD.EXTOP extended operation routine, which temporarily replaces the channel ID in A0 with the address of the channel details. SD.EXTOP sifts out non-display channels, reporting 'bad parameter'.

The FIND_WINDOW subroutine works out the location and storage requirements for the window. By the time it has finished, A1 points to the first word of the window in display memory, the amount of space needed is in registers D1 and D5, with the width of each window line in D0 and the number of lines in D4.

The width is measured in sixteen bit words, with one subtracted from D0 and D4 to suit the DBRA loop instruction, which counts down from the initial value to -1. This fiddling around is justified because it simplifies the keyword routines, which all use FIND_WINDOW.

The next step is to allocate memory for a stored image. MT.ALCHP attempts to reserve D1 bytes on the Common Heap. If this works, the space is prefixed with 'Bufa', so that the DIY Toolkit DISCARD command knows that it can safely be released. The space is also added to the list maintained by Toolkit 2, so it can be released with RECHP or CLCHP, and automatically freed after LOAD, LRUN or NEW.

Linked list

Toolkit 2 uses the first four bytes of each heap allocation made with ALCHP to link the allocations into a list. The undocumented system variable at 224(A6) in the SuperBasic area normally holds zero, but Toolkit 2 updates this to point to the link in the most recent ALCHP space.

In turn, this link may contain the address of another link, at the start of another ALCHP space, or zero to signify the end of the list. CLCHP and the commands that remove the current program go through the list and deallocate each space. The list is ignored if you do not have Toolkit 2 on your machine. The next eight bytes indicate the size of the stored image. First comes a long word, the total size requested in bytes, followed by two words: the number of pixel lines in the window (less one to suit DBRA), and the compression indicator held in D6. All the remaining space is used to store window pixel information, read from the screen display.

Shared code

The subroutine labelled SCAN_LINES is central to all DIY Windows keywords. My original keyword routines each used a double loop to scan the window and process each word of display memory. The outer loop stepped from line to line, while the inner loop processed the individual words of each line. This was rather a waste of memory, as the same instructions were used to step between

```

define      dc.w      5          5 new PROCedures
            dc.w      show-*
            dc.b      6,'W_SHOW'
            dc.w      swap-*
            dc.b      6,'W_SWAP'
            dc.w      swap-*
            dc.b      6,'W_SWAP'
            dc.w      setred-*
            dc.b      7,'SET_RED'
            dc.w      setgreen-*
            dc.b      9,'SET_GREEN'
            dc.w      0          End of PROCs
*
            dc.w      2          2 new FuNctions
            dc.w      store-*
            dc.b      7,'W_STORE'
            dc.w      crunch-*
            dc.b      8,'W_CRUNCH'
            dc.w      0          End of FNs
            end

```

lines in every case.

The improved version is almost as fast, yet 48 bytes shorter, because the shared subroutine SCAN_LINES is used to step between lines. The action performed on each line is determined by the value in register A3 which points to a distinct subroutine for each possibility. In every case D2 indicates the line width, and A4 points to the first word of each line.

The line manipulation subroutines are collected at the end of the listing. For instance, SAVE_BYTES copies alternate bytes from display memory, at A4, to the buffer at A0. SAVE_BYTES is used by W_CRUNCH. SAVE_WORDS is the corresponding loop used by W_STORE; it copies a word at a time. SWAP_WORDS is used by W_SWAP and W_SWOP; it exchanges odd and even bytes in display memory.

All eight routines are called from SCAN_LINES, which sets D2 and A4 to process each line in turn. The code could be 28 bytes shorter if the DBRA D2 loops were moved to SCAN_LINES, but this would slow things down disproportionately because of the need to call a subroutine to process each word, rather than each line.

Wide windows

Minerva users with wide windows might speed up W_STORE and W_SHOW by calling the MM.MOVE vector, \$158. The advantage is slight as they must call the vector for each line, unless the window is a full 512 pixels wide. The remainder of the code for W_STORE and W_CRUNCH should be familiar. This 'normalisation' code is used by many QL extensions. It was discussed in my May 1988 QL World column and CHANCODE_DOC in DIY Toolkit Volume C. It converts the address of the heap allocation in D1 into six-byte QL floating-point form, so it can be returned to Basic.

You should now be sufficiently cognizant of the DIY Windows design to understand the entire SWAP routine. Only

ten lines of unique code are needed to implement W_SWAP and W_SWOP. The first five lines call GET_CHANNEL and FIND_WINDOW to read and analyse the channel parameter. Then they point A3 at the SWAP_WORDS subroutine, before falling into the SCAN_LINES loop, which calls SWAP_WORDS for each line in turn.

SET_RED and SET_GREEN work much the same way, but take an extra parameter which determines the routine called by SCAN_LINES. If this parameter is 0 or 1 the corresponding colour bytes must be set or cleared. In both cases the FILL_BYTES loop is used to copy the value of D6 into alternate bytes of the window. D6 is zero if the bytes are to be cleared, or -1 to set all bits.

If the second parameter is -1 the RED_BYTES or GREEN_BYTES loop is selected, to copy information from one set of colour bytes to the other. Green bytes are stored before red ones, so SET_RED -1 goes faster than SET_GREEN 1, which must step backwards to copy each red byte to a green location, then step forward to the next word.

The only unexplained keyword is now W_SHOW, which is encoded after the FIND_WINDOW subroutine. Unlike the others, this takes a long integer parameter after the channel number. This is the address of the window contents to be restored. SHOW starts by checking that there are two parameters, and hides the second one before calling GET_CHANNEL. There is no need to check the value of D3 after this call as A3 and A5 differ by 8, delimiting a single parameter.

CA.GTLIN is used to fetch the long word address, which should be positive and even. The compression indicator is read into D6 before the call to FIND_WINDOW, so it gets the size right. W_SHOW reports 'out of range' if the size or depth of the window has changed since it was stored. Otherwise it uses LOAD_WORDS or LOAD_BYTES to restore the image, depending on the value of D6.

Various modes

DIY Windows commands are designed to suit Mode 4 displays, but they can also be used to save and restore Mode 8 windows. There is a potential problem with SET_GREEN and W_SWAP in Mode 8, as these commands may set flash bits as well as green bits, leading to a chaotic flashing display.

The problem occurs because Mode 8 uses four bits to record the details of each pixel. Alternate bits in the even bytes turn flashing on and off. Corresponding bits in the odd bytes control the blue component of the colour, so the odd-numbered ink colour blue, magenta, cyan and white make the window flash after SET_GREEN or W_SWAP.

To correct this, alter the commands to check the display mode with a call to MT.DMODE (TRAP #1, D0=16); remember to preserve A4 as it is corrupted by the call. If Mode 8 is in use you must mask each byte stored at an even address with AND #170 to ensure that the flash bits are not set. This enhancement requires two new line-scanning loops, based on GREEN_BYTES and SWAP_WORDS.

The existing code works perfectly in the Thor XVI's extra Mode 12, which replaces the flash bits with intensity controls, giving 16 colours on an RGBI TTL monitor. It also copes with Minerva's second screen, as it works out the window address from SD.SCRB in the channel definition.

Extra screens

In fact it is possible to have even more 'logical screens' in QL memory. Sinclair's display hardware can only read screens starting at 131072 or 163840, but Minerva lets you move the system variables even higher in memory, leaving space for extra 32K screens. To display these you need to move their contents into one of the hardware display areas, with MM.MOVE or W_STORE and W_SHOW.

If you already read the screen from a remote machine using the MEM device and network file server, all you need to do is change the start address used by MEM. The Minerva rom can put text anywhere in memory, depending on the values in the channel definition. If you have a Gold Card, scrolling and output to screen addresses above 196607 should be extra-fast as there is no need to update slow eight bit memory.

There are two ways to re-direct text and graphics to extra screens. You can update SD.SCRB to point to the start of the required screen, at any even address, or leave the default of 131072 and store a top line number greater than 255 in SD.YMIN.

This value is multiplied by 128, using word arithmetic, and added to SD.SCRB to find the base address of the window. Thus it is possible to position a window that overlaps between two screens! I thank reader PHTanner of Glasgow for this idea.

Such POKEs require the use of CHBASE, from DIY Toolkit Volume Q, to find the channel definition from Basic. Machine coders can get the same effect with an SD.EXTOP call.

Next issue

DIY Toolkit will be back next month with a useful command that will help you generate neat formatted reports on the vocabulary of your SuperBasic interpreter, on all QL systems from AH to Minerva, occupying only 276 bytes of memory. I am always on the lookout for new, useful and concise extensions to the Qdos system, and welcome suggestions from readers. Please write to DIY Toolkit, care of QL World.

SUBSCRIBE TO SINCLAIR QL WORLD!

SUBSCRIPTION FORM

Rates (12 months)

U.K. **£23.40**
Europe: **£32.90**
Rest of World **£40.90**

Select back issues available on request. Please call for details.

Cheques should be made payable to Arcwind Ltd.

Please debit my credit card/Access/Visa

Card No. _____

Expiry Date _____

Please send me a 12 Month subscription to Sinclair QL World

Name _____

Address _____

Telephone No. _____

Signed _____

Date _____

Send the completed form to:

Arcwind Ltd

The Blue Barn,
Tew Lane
Wootton
Woodstock
Oxon OX7 1HA
Tel: 0993 881181
Fax: 0993 811481

SOFTWARE FILE

INFORMATION

Program: *Open World 2.0*
Supplier: Ergon Development,
 c/o Davide Santachiara, Via
 Emilio de Marchi n.2, 42100
 Reggio Emilia, Italy.
Memory: 256K
Price: £25(disk) plus £5 carriage.
 All payable to D. Santachiara.
 Add £4 for sterling cheques;
 nothing for Eurocheques in Lire.

Open World

 **Rich Mellor looks through a collection of pictures which can be easily converted for use by and on a QL.**

The QL has various programs which can be used to design pictures both for use on the computer screen and for use in printed material. However, for the less artistic, it can be difficult to come up with ideas. Other computers have a wealth of public domain picture files which can be used, some of which are truly excellent pieces of computer-generated art. These can be used in a display or even to improve the look of your documents.

Although there are some public domain pictures for the QL, these tend to be what is known as 'clip art' (a series of small pictures stored on a larger screen, which may then be cut out from that screen and used to embellish desktop publishing). There have however been two new releases on the QL scene to give the user, greater access to pictures designed on other computers. One of these is CGH Service's *SToQL*, which allows you to access various types of pictures created on an ST computer (a program which I will not dwell upon because I am partly responsible for it). *Open World* is the other.

Open World comes supplied on a disk or microdrive together with a twelve page A4 manual. The manual is well written and easy to follow. On loading the program with the supplied boot program, you are presented with a start-up screen giving a little information about the program. Pressing any key will give you the main menu from where you can decide the type of screen to be converted into QL format.

Provided that you have already used a utility (such as *Xover*) to convert a disk containing the screens into QL format, *Open World* can then be used to read the

disk and convert the following screen formats:

IFF - Generated on an Amiga computer (for example using the program *Deluxe Paint*),

GIF - A standardised screen format widely used on PC's,

TIF - Generated by different scanners and desktop publishing programs on PCs and Macintosh computers,

CUT - A less widely used format but available on various computers.

Open World uses a system of various easy to operate menus to choose the various functions and set various parameters. The menus are based upon Ergon's own standard menu system, which allows you to choose options either by moving a cursor bar onto the appropriate item, or by pressing a key associated with each option. Their operation is explained in the manual and, as with all of Ergon's programs, this makes the program easy to use.

Once you have chosen the type of image to convert, a menu is displayed containing a list of the files on the given Working Device which contain the appropriate suffix (eg _GIF for GIF files). You can then easily choose which file you wish to convert, using the cursor bar. If you are lucky enough to have the Files-2 system, you will be pleased to note that *Open World* supports full sub-directories.

If you have chosen to convert a GIF or IFF file, a small window is opened

on screen which displays useful pieces of information about the file, which can be used to improve the image if you wish.

Having loaded the file to convert, you are given a further sub-menu which allows you to set the conversion parameters. Some converted screens may be larger than the QL screen, so to help with these, you are able to specify both x and y offsets for the top left hand corner of the screen being loaded. On top of this, you can also alter the size of the display by altering the parameter 'Output scr_500x240a6x8' to the desired window size. Other parameters allow you to specify whether the screen should be converted in mode4 or mode8, and how much memory should be allocated for the conversion of GIF files.

Another option (called Dithering) allows you to convert screens in monochrome mode. *Open World* uses quite a complex algorithm to ensure that the resultant black and white image retains the original resolution and the grey scales introduced make a fair attempt at retaining the colour diversity. This does however tend to degenerate mode8 pictures, although this will of course depend very much upon the original image.

Having set the conversion values, you can then choose to process the file. *Open World* will load the appropriate conversion routine from the Main Device prior to loading the file. The conversion process is generally quite quick, although the GIF conversion takes longer for various reasons explained in the manual. Apparently, GIF files can be 'interlaced' and contain up to 256 colours. As a result, the image must be loaded into memory and then some calculations must be carried out upon the image before it can actually be displayed on screen.

Although the QL cannot (currently) display as many as 256 colours (even the THOR XVI can only display a maximum of 16 colours), the conversion programs each make a valiant attempt at ensuring that the picture retains most of its original resolution. However, with current levels of QL technology, you will need to view the image on a black and white monitor to obtain the best results!

Having converted the program you can then save the image in QL format onto any device, thus allowing you to use these pictures in your own programs or displays. If however, you have several pictures to convert, it may be useful to use the 'Transfer to ram' option (provided that you have a ramdisk available). If you choose this option, *Open World* will transfer all of its overlays onto a ramdisk (ram1_) and from then on it will access the ramdisk for its own files, allowing you to use the program easily on a QL with only one disk drive.

Overall the program is very slick, and enables you to convert the various form of images very quickly and easily. I only found one or two minor errors in the program, all of which have been notified to Ergon and which I understand they will soon remedy. If you are in need of a few images to get you started, I understand that Ergon can also supply two disks (for £2 each) full of GIF and IFF screens. I have seen a few of these screens and can say that they would appear to be of very good quality. Many users will undoubtedly find a use for such a program, particularly those with access to bulletin boards, which seem to contain an endless stream of such images.

SOFTWARE FILE

Qmenu

INFORMATION

Program: QMenu

Supplier: Jochen Merz Software

Im Stillen Winkel 12

4100 Duisberg

West Germany

Price: £12.00 plus VAT, plus £4.50 carriage

Updates cost £5 plus £4.50 carriage.

Qmenu helps programmers use parts of the QJump Thing system without mastering the entire system. Ian Bruntlett assesses it.

Qmenu is the documentation for an extension Thing (special toolkit), menu_ext, which provides easy-to-use menus and a special scratch-pad area. Menu_ext defines two extension things, 'Menus' and 'Scrap'.

The advantage of writing a toolkit as an extension Thing is that it is not specific to SuperBasic, and may be used from other languages such as C or Forth. Most normal QL toolkits can only be used from SuperBasic. All the work required to handle Things in your QL is handled by the Thing System in the Qjump file 'hot_ext', which is bundled with Qpac I and II, Qd3, DataDesign, etc. The Thing system is a (linked) list of named resources that may be used by QL programs in an ordered manner.

Qmenu provides enough information to use the 'Menus' and 'Scrap' extensions without having to worry about how to use extension Things. Things are given a technical explanation in chapter 17 of the *Qdos Reference Manual* available from Jochen Merz software for £30 excluding postage.

Expanded QLs are beginning to see more and more extensions building upon other extensions. First seen in Qram, PTR_GEN handles the pointer and added handling for extra graphic objects, while WMAN built upon PTR_GEN to provide an easy-to-use menu system. Programmers find WMAN hard to use from their programs, as they have to wrestle with Qptr, the documentation for PTR_GEN and WMAN. Hot_ext, introduced in Qpac I, contains the Thing system and the Hotkey System II.

Co-operation

'Menus' follows this theme of co-operation, using WMAN and Hot_ext. The 'Menus' extension uses the Thing system (hot_ext) to let the rest of the system know that it is 'in the house'. All a program has to do is to ask the Thing system if it can use the Thing called 'Menus'. To provide the easy-to-use, pointer-driven menus, the 'Menus' Thing uses the Window MANager, WMAN.

A SuperBasic program demonstrating the uses of Qmenu

```

100 REMark QmenuQLW2_bas (C) Ian.R.Bruntlett April 1991, 1992
110 REMark demo of new qmenu, SELECT_LIST
120 REMark 5/5/92 modified for more useful file picking
130 _Compiled%=1
140 REMark $$asmb=flp1_Outln_ext,0,54
150 REMark $$off
160 _Compiled%=0
170 REMark $$on
180 IF _Compiled% THEN OPEN #0:"con_512x256a0x0":OUTLN #0:512,256,0,0
190 :
200 REPEAT Loop
210 Option=SELECT_LIST("SLST demo","1. Stuff filename/2. Stuff file from dev/3. Stuff device/4. Execute
job/5. VIEW a file/6. Set PROGD$/7. Edit PROGD$/8. Set DATAD$/9. Edit DATAD$/A. Edit stuffer",,10)
220 SELECT ON Option
230 =0:PRINT #0:"ESCAPE!":EXIT Loop
240 =1:
250 file$=FILE_SELECT$("Stuff filename")
260 IF file$<>" THEN HOT_STUFF file$:EXIT Loop
270 =2:
280 _Hk$=FILE_SELECT$("filename to stuff",,DIR_SELECT$("directory for stuff",,2),,2):IF _Hk$<>" THEN
HOT_STUFF _Hk$:EXIT Loop
290 =3:
300 _Hk$=DIR_SELECT$("directory to stuff",,2):IF _Hk$<>" THEN HOT_STUFF _Hk$:EXIT Loop
310 =4:
320 file$=FILE_SELECT$("Execute job",,PROGD$,,_exe')
330 IF file$<>" THEN EXEC file$
340 =5:
350 file$=FILE_SELECT$("Pick a file to view")
360 IF file$<>" THEN VIEW_FILE file$
370 =6:
380 subd$=DIR_SELECT$("Set programme default")
390 IF subd$<>" THEN PROG_USE subd$
400 =7:
410 newPROGD$=READ_STRING$("Edit programme default",PROGD$)
420 IF newPROGD$<>" THEN PROG_USE newPROGD$
430 =8:
440 subd$=DIR_SELECT$("Set data default")
450 IF subd$<>" THEN DATA_USE subd$
460 =9:
470 newDATAD$=READ_STRING$("Edit data default",DATAD$)
480 IF newDATAD$<>" THEN DATA_USE newDATAD$
490 =10:newSTUFF$=READ_STRING$("Edit stuffer buffer",HOT_NAMES(" "))
500 IF newSTUFF$<>" THEN HOT_STUFF newSTUFF$
510 END SELECT
520 END REPEAT Loop
530 STOP

```

To keep things simple I will call a menu from Qmenu a Qmenu, and any other menu a menu. When I give a Qmenu's SuperBasic name I'll put its four character 'extension thing' name in square brackets after it.

First of all, are the Qmenus practical? Yes. When a user is confronted with a Qmenu, it is no different to being given a menu from Qram

or Qpac II. This is not too surprising, as 'Menus' uses WMAN (which is also used by Qram and Qpac II). It does show that Jochen Merz has observed the few guidelines in Qptr which are along the lines 'CTRL F2 causes the information displayed to be refreshed, ESCAPE removes a window without performing any (further) action, etc.'. So there are no nasty surprises for

the user.

'Menus' is a set of nine specialised menus biased towards file handling. The colours of the menus can be defined by setting 'colourways', which are a number from 0-4, the default being 'white/green'. As the menus rely on the window manager, the programmer must ensure that the job the Qmenu is being used in has a 'managed window' (set by iop.outl, trap #3, d0=\$7a). SuperBasic programs are provided with OUTLN, an extra toolkit command which does this.

The most impressive of the Qmenus is FILE_SELECT\$[FSEL]. This brings up a large menu which has two large areas, one to show subdirectories (but only on the Miracle hard disk and other level-2 device drivers), and the other to show a list of filenames. It is very easy for the user to change device - a list of the directory devices present in the QL is part of the menu, choosing one of them causes FSEL to display a list of filenames from the new device. There is a slight problem with the list of devices - if there are two devices that begin with the same letter then only the first of those devices can be chosen with a single key press. This is only a minor problem, as keyboard users can still move the pointer to the desired device name and 'hit' it.

Hitting

'Hitting' the sub-directories entry pulls down a further qmenu - DIR_SELECT\$ [DSEL]. This qmenu allows you to choose a device from 'mdv1/2, flp1/2, ram1/2, win1/2', or any of the eight user defined sub-directories that may be set with Qjump's Config. This shows a weakness of Config - there is no 'update' function: if you configure a piece of software and later on receive an updated version, all you can do is to

overwrite the old version and configure the new version. It would be much more convenient if I could just tell Config: 'Here is the old configured version, use my configuration choices from the old version as the default choices for the new version'.

'Hitting' the file extension entry pulls down another qmenu - EXT_SELECT\$[XSEL]. This is a list of ten common file extensions. They are common to programmers - 'bas\pas\asm\txt\c\err\lis\t87\map\exp'. The default extensions may be configured by the user to suit.

The FILE_SELECT menu has one more trick up its sleeve - it can read the stuffer buffer (ALT-Space) so the filename return could also be the current or the previous contents of the stuffer buffer.

There is one refinement I'd like to see in FILE_SELECT. I would like it to be able to show a file (for example, with the Qmenu VIEW_FILE) so that the user can make sure a certain file is the correct file before clicking 'Ok'.

FILE_ERROR will either return -1 if the error to be dealt with is not a file error, or it will produce a relevant menu with options for retry/abandon/overwrite, etc, returning a unique value for each user choice. This is a new addition to Qmenu.

VIEW_FILE [VIEW] is a relatively simple file viewer. It shows a file in a big window and allows you to set WRAPON (lines are wrapped around at the end of the window, like COPY file TO SCR) or to set WRAP OFF (lines too long to fit in the window are truncated, like VIEW file). It is similar to the Qpac2 file viewer - pressing Space reads a further line, Enter reads a whole page and CTRL-F2 (Wake) causes the beginning of the file to be displayed again. The only missing facility is a scroll bar to move quickly from one part of the file to another.

Config

New versions of Qjump's Config now use the FILE_SELECT menu if it is present. Now you have the strange situation of being able to run Config, which in turn will use menu_text to get a filename from the FILE_SELECT menu, which in turn gets configured by Config! This kind of co-operation should be more frequent.

The other qmenus are less file-oriented:

READ_STRINGS [RSTR] brings up a small window with a title and prompt and allows the user to edit some text (possibly a file name).

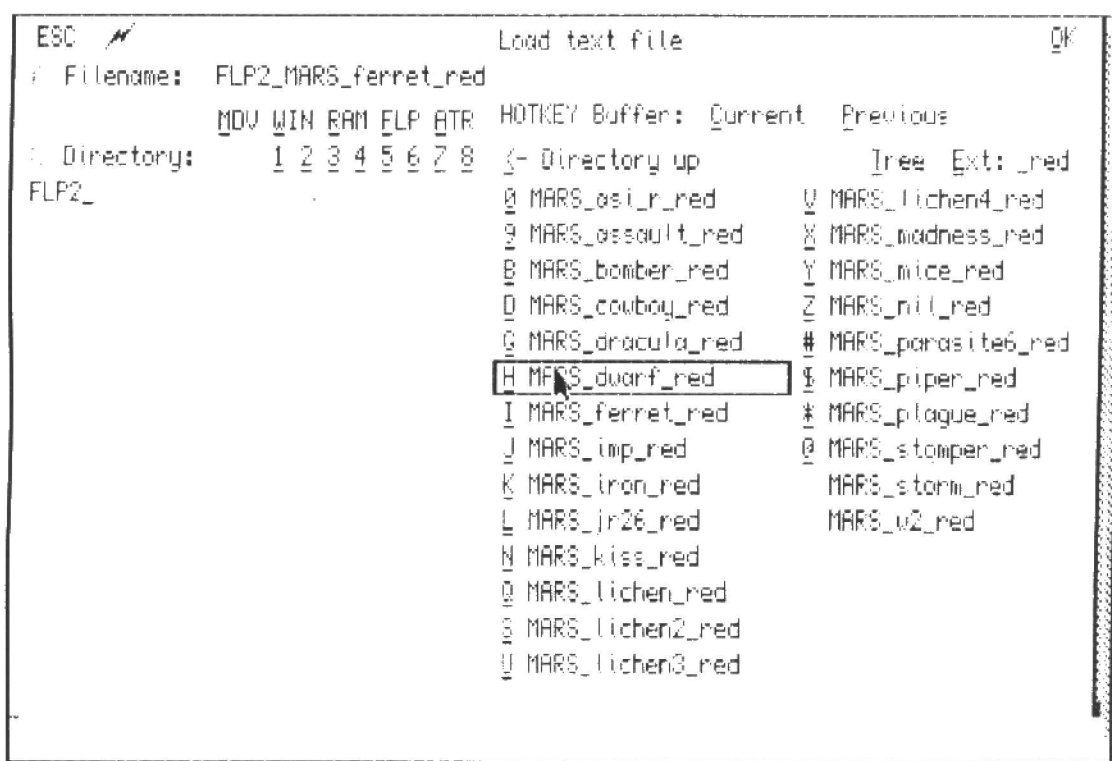
REPORT_ERROR [RPER] brings up a small window with a standard Qdos error report in it and waits for the users to acknowledge its existence - then it removes itself.

ITEM_SELECT [ITSL] brings up a small window with a title, prompt and up to three options to be chosen. Despite having three options, this menu can still be used to display a Toolkit 2 look-alike 'yes/no/all/quit' prompt as it returns 1 to 3 for a selected option and 0 for ESCape. The options may be chosen by pressing the first letter of the option name or by clicking an option with a mouse.

Choice%=item_select('QLW Test menu','You have no option but to choose one','No','All','Yes')

During the passage from Qmenu version two to version three, Qmenu has diversified into list handling. There are three list handling routines - SELECT_LIST, SELECT_LISTS and LIST_SELECT. Only LIST_SELECT is meant to be used these days - the SELECT_LIST routines are only included for compatibility.

The LIST_SELECT routine will bring down



a menu of items, but has two modes of operation. It can be used as an option list where the user can switch options on/off, returning when the user has finished setting the options. Or it can act as a conventional menu, returning when the user has selected a single option.

The other extension Thing, 'Scrap Extensions', handles the 'Scrap'. The 'Scrap' is an area of memory which can be used for memory-based importing/exporting between programs. The programs have to specifically access the scrap, so it won't be used by the Psion programs.

There are SuperBasic commands and machine code routines to clear the scrap, put information in it, find out what kind of information is in the scrap and to read the scrap.

Documentation

So now you have been introduced to 'Menus', what about the documentation and facilities for programmers? In no way do the above descriptions render the 'Qmenus' documentation redundant. The manual seems to have been written in the 'Qptr' style of manual writing, that is, concise and to the point. Occasionally it is too concise - the assembly language programmer would feel a bit let down if all there was to go on was the manual. Fortunately, there is a full demo in assembler on the disk, and the source files are provided.

To access the 'Menus' from assembler, Jochen Merz provides some assembly language source code files. Previous versions of Qmenu provided S-ROFF (Sinclair Relocatable Object File Format) files that needed a compatible assembler. Now that the source code for the access routines is provided, you can use Qmenu with any QL assembler.

If you are unfamiliar with the 'Thing' system, you should read the Qjump document 'thing_ext' on

the Quanta SPECIALS_0 disk before using Qmenu. You may find my 'Qdos_Listthg_bas' on the CGH Services MISC PD disk a handy way of finding information on Things.

Despite those reservations, Qmenu is definitely usable by assembly language programmers. Each menu is defined reasonably well, although the manual occasionally returns to being too concise. For instance, the FILE_SELECT menu can modify the directory string or extension string passed to it. This is explained away with "If the update bit in the directory or extension parameter is set, then, if the user changes the extension or the directory, the string is updated (if it is not a key)". Which update bit? Could it be in the definition of the Thing parameters in 'Menus' or does it refer to the parameter block that we pass to the qmenu code

itself?

The SuperBasic programmer fares better than the assembly language programmer as far as documentation goes. Each SuperBasic extension is listed and defined. At the moment the 'SuperBasic-Thing' interface cannot handle multiple return parameters, so one or two facilities are unavailable from SuperBasic. It even mentions some facilities that are due to be implemented.

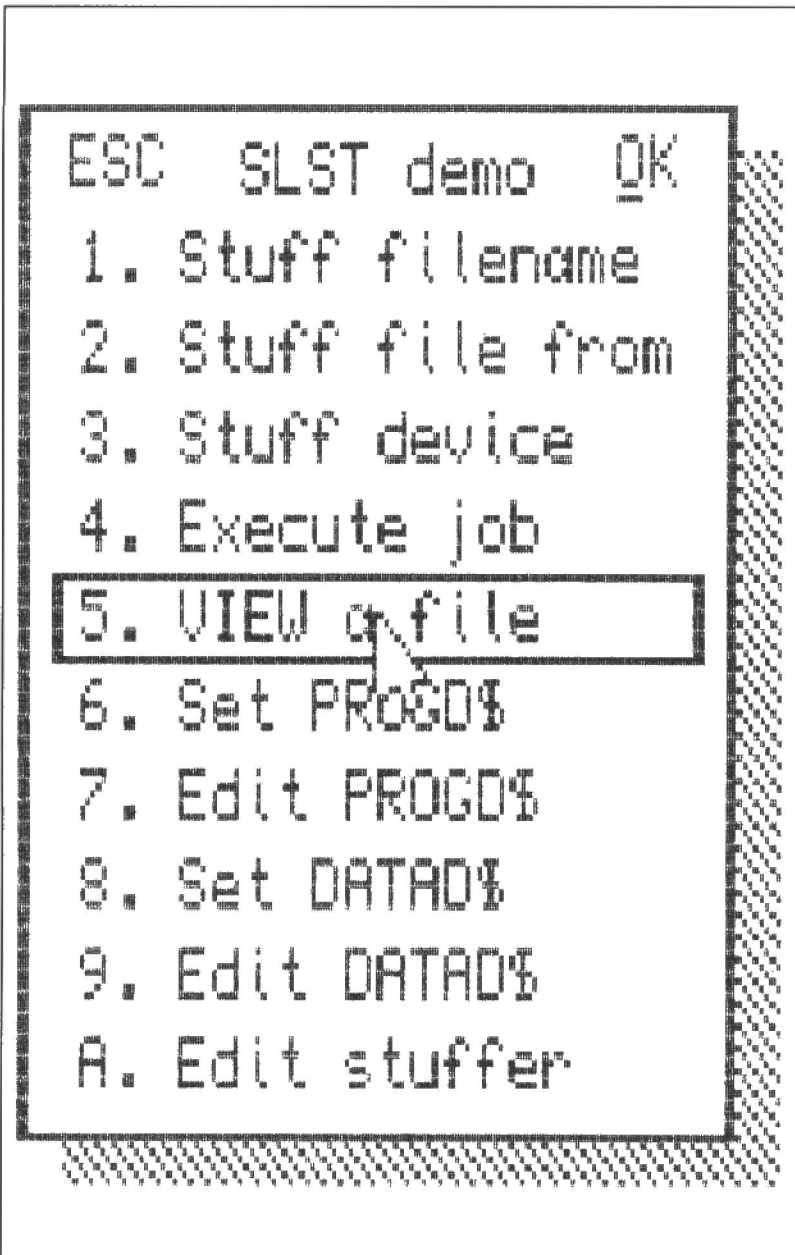
Serious

If you are a programmer who is going to make serious use of qmenus within your own programs, buy Qmenu. You are going to need it. A dabbler could get by on guesswork without Qmenu, but would end up not using the full facilities of the 'Menus' extensions.

The 'Menus' extensions are very useful and easy to use in a program. If you have a small program which would be very useful but the user interface work via Qptr would be too much effort, use the Qmenus. You won't have the degree of control that Qptr gives you, but the result will be a saving in time. All that is missing now from Qmenu is a 'main window' menu which has individual 'Qmenus' built into it, placed by the programmer, waiting to be invoked.

I have two simple SuperBasic programs which have been made easier to write and better to use by Qmenu. One, 'vdat', is a filter that uses SELECT_LIST when deciding which Psion printer driver to list. The other, a test bed for this article, is on a hotkey to set Toolkit 2 defaults from the DIR_SELECTS, stuff a filename from FILE_SELECT\$, etc. Both were simple programs made more useful by Qmenu.

People who are going to use the 'Menus' extensions only in packages that use it (such as Easy Pointer or Qd2/3) will not gain anything by buying Qmenu.



```

ESC  SLST demo  OK
1.  Stuff filename
2.  Stuff file from
3.  Stuff device
4.  Execute job
5.  VIEW a file
6.  Set PROGD$
7.  Edit PROGD$
8.  Set DATAD$
9.  Edit DATAD$
A.  Edit stuffer
    
```

Systematic Machine Code Programming

In part 9 of this series for new programmers, Alan Bridewell puts his routines to the use of saving and re-loading screen files.

Any program which is concerned with screen graphics will usually require the ability to save screens to files, and also reload such screen files back to the screen. In this section, we shall look at routines for doing just that. Most of the coding here has been dealt with before as smaller chunks, so if you have been following the series, you should not have much difficulty with it. However, as they occur so often in applications, it seems reasonable to include them in the series in their own right, rather than leave you to construct them from smaller chunks.

Listing one gives the Screensave routine. Although it is not complicated, it does present a particular problem which needs to be overcome one way or another. The problem is this: in order to save the screen to a file, we need to open a new file on disk or microdrive. To do this, we need to first prompt for a file name, which must then be typed in. This, of course, means using the screen, which is also to be used for the display to be saved on the file. One solution would be to use a window in part of the screen not required for the display, which is fine so long as the display does not require the entire screen.

I prefer a slightly different solution. I open a suitable file before

Listing 1

```

; *****
; 'SCREENSAVE'
; *****
; THIS ROUTINE WILL PROMPT FOR THE NAME OF A NEW FILE, AND SAVE THE QL'S
; SCREEN RAM IN THE FILE
; PUT PROMPT ONTO SCREEN
;
.MESS          MOVE.L    (A7),A0      ; OR 4(A7),A0 OR 8(A7),A0, ETC.
;                                     ; CHANNEL ID IN A0
          LEA.L    MESSAGE,A1      ; BASE ADDRESS IN A1
          MOVE.W   $D0,A2          ; UT_MTEXT VECTOR IN A2
          JSR      (A2)
          BRA.S    INSTR           ; SKIP MESSAGE
.MESSAGE       DC.W      44         ; LENGTH OF MESSAGE
          DC.B     'Name of new screen file'
          DC.B     10              ; CR,LF
          DC.B     '(including device): '

; GET STRING FROM KEYBOARD
;
.INSTR         MOVEQ    #BUF_LEN,D2  ; LENGTH OF BUFFER IN D2
          MOVE.W   #FFFF,D3        ; INFINITE TIMEOUT
          LEA.L    BUFFER,A1        ; BASE ADDRESS OF BUFFER IN A1
          MOVEQ    #2,D0           ; #IO_FLINE IN D0
          TRAP     #3
          LEA.L    BUF_POS,A0       ; BUF_POS IN A0
          SUBQ.L   #1,D1            ; SUBTRACT 1 FROM THE D1 REMOVES
;                                     ; THE LF FROM THE STRING COUNT.
          MOVE.W   D1,(A0)         ; PUT NEW STRING LENGTH IN BUF_POS

; TRY TO OPEN A CHANNEL TO THE FILE
;
          LEA.L    BUF_POS,A0       ; BUF_POS IN A0
          MOVEQ    #2,D3           ; OPEN NEW EXCLUSIVE FILE
          MOVEQ    #-1,D1          ; JOB ID FOR THIS JOB
          MOVEQ    #1,D0           ; #IO_OPEN IN D0
          TRAP     #2
          TST.L    D0              ; ERROR?
          BEQ.S    CONT           ; IF NOT, THEN CONTINUE. ELSE:-

; IF ERROR, THEN THIS IS USED
          BRA      MESS            ; GO BACK TO PROMPT
; ALTERNATIVELY, YOU COULD BRANCH TO CODE TO KILL THE JOB
;
; IF NO ERROR, THEN THIS CODE IS USED.
; ( OR YOU COULD JSR TO A ROUTINE TO GENERATE GRAPHICS,
; AND RTS TO .TRA )
.CONT         MOVE.L    A0,-(A7)    ; SAVE FILE CHANNEL ID
.MESS2        MOVE.L    4(A7),A0    ; CHANNEL ID IN A0
          LEA.L    MESSAGE2,A1      ; BASE ADDRESS IN A1
          MOVE.W   $D0,A2          ; UT_MTEXT VECTOR IN A2
          JSR      (A2)
          BRA.S    PRIORITY        ; SKIP MESSAGE
.MESSAGE2     DC.W      29         ; LENGTH OF MESSAGE
          DC.B     'NOW CREATE SCREEN TO BE SAVED'
          DC.B     10              ; CR,LF

; REDUCE THE PRIORITY TO 1
.PRIORITY     MOVEQ    #B,D0        ; #MT_PRIOR IN D0
          MOVEQ    #-1,D1          ; OF THIS JOB
          MOVEQ    #1,D2           ; TO 1
          TRAP     #1

; MONITOR THE KEYBOARD FOR <CTRL><SHIFT>X
.KEYROW       LEA.L    KEYR,A3      ; KEYROW PARAM TABLE IN A3
          MOVEQ    #11,D0          ; #IPCDM IN D0
          TRAP     #1
          CMPI.B   #11,D1          ; IS <CTRL><SHIFT>X PRESSED?
          BEQ.S    PRIOR2          ; IF PRESSED, MOVE ON
          BRA.S    KEYROW          ; IF NOT, TEST AGAIN

```


setting up the screen to be saved. Then when the screen is ready, pressing the appropriate key combination will save the screen to the file.

Two ways round

This method lends itself to two alternatives. If the same program is going both to generate and to save the display, it must branch off to the routines which generate the display after the file has been opened, and then return to the routine when the display is ready to be saved. On the other hand, the screen-saving routine can be in a separate, multitasking program. In this case, once the file is open, the routine can reduce its priority to 1 so that it does not slow down the screen generating program, and simply wait in the background monitoring the keyboard for the right combination of keys.

At this point, the user can Ctrl-C from the saving program into the generating program. When the appropriate key combination is pressed, the saving routine will increase its priority to 32 again, and save the screen.

Since a lot of the routine is given over to this approach, it is the default setting. The alternative is there as a one line BRA statement.

The routine starts by putting a prompt into a suitable screen channel which was previously opened. It then takes a string from the keyboard as a channel name, and tries to use it to open a new, exclusive channel. If it is unsuccessful, it loops back to repeat the prompt. If the channel is successfully opened, its channel ID is saved on the stack, and a prompt is put on the screen to carry on and generate the screen which is to be saved.

After this, the routine reduces the priority to one, and then goes into a loop waiting for a particular combination of keys to be pressed. The default setting is to test for Ctrl-shift-X, because the three keys are in the same keyrow, and it is an unusual key combination. Clearly, you can alter this to suit your own requirements. Once that key combination has been pressed, the priority is raised again to 32, and the transfer of bytes from screen ram to file goes ahead. Finally, the file has to be closed.

Taking it apart

If you are interested in taking this apart, you will see that it is almost entirely constructed of smaller chunks from earlier parts in this series. They are, in order Message, Instr, Open, Message, Priority, Keyrow, Priority, Close. The only new bit comes just before the final Close. This is where

```

;
; KEYROW PARAMETER TABLE
.KEYR      DC.B      9      ; READ KEYROW
           DC.B      1      ; ONE PARAMETER
           DC.L      0      ; LOWEST FOUR BITS ONLY
; ALTER NEXT PARAMETER TO REQUIRED KEYROW
           DC.B      7      ; TEST ROW 7
           DC.B      2      ; REPLY OF 8 BITS IN D1
;
; INCREASE PRIORITY BACK TO 32
.PRIOR2    MOVEQ     #$B,D0      ; #MT_PRIOR IN D0
           MOVEQ     #-1,D1      ; OF THIS JOB
           MOVEQ     #32,D2      ; TO 1
           TRAP      #1          ;
;
; TRANSFER BYTES FROM SCREEN TO FILE
.TRA       MOVE.L     (A7),A0      ; FILE CHANNEL ID IN A0
           MOVE.L     #$20000,A1   ; START ADDRESS OF SCREEN
           MOVE.W     #$FFFF,D3    ; INFINITE TIMEOUT
           MOVE.L     #$8000,D2    ; SCREEN SIZE IN D2
           MOVEQ      #$49,D0      ; FS.SAVE IN D0
           TRAP      #3
           MOVEQ      #2,D0        ; #IO_CLOSE IN D0
           TRAP      #2
;
; *****
; TO BE POSITIONED AFTER THE CODE
; WE NEED 'BUFFER' FOR THE STRING AND 'BUF_POS' FOR THE STRING COUNT
;
.BUF_LEN    EQU      50          ; LENGTH OF INPUT BUFFER
;
.BUF_POS    DC.W      0          ;
;
.BUFFER     EQU      *          ;
;
; ***** NOTE ***** BUFFER SHOULD COME LAST OF ALL.
;
; *****
Listing 2
; *****
; 'SCREENLOAD'
; *****
; THIS ROUTINE WILL PROMPT FOR THE NAME OF A SAVED SCREEN, AND LOAD THE
; NAMED FILE INTO THE QL'S SCREEN RAM
;
; PUT PROMPT ONTO SCREEN
.MESS      MOVE.L     (A7),A0      ; OR 4(A7),A0 OR 8(A7),A0 ETC.
           LEA.L      MESSAGE,A1   ; CHANNEL ID IN A0
           MOVE.W     $D0,A2      ; BASE ADDRESS IN A1
           JSR        (A2)        ; UT_MTEXT VECTOR IN A2
;
.MESSAGE    BRA.S     INSTRING     ; SKIP MESSAGE
           DC.W      46          ; LENGTH OF MESSAGE
           DC.B      'Name of saved screen file'
           DC.B      10          ; CR,LF
           DC.B      '(including device): '
;
; GET STRING FROM KEYBOARD
.INSTR      MOVEQ     #BUF_LEN,D2  ; LENGTH OF BUFFER IN D2
           MOVE.W     #$FFFF,D3   ; INFINITE TIMEOUT
           LEA.L      BUFFER,A1   ; BASE ADDRESS OF BUFFER IN A1
           MOVEQ      #2,D0        ; #ID_FLINE IN D0
           TRAP      #3
           LEA.L      BUF_POS,A0   ; BUF_POS IN A0
           SUBQ.L     #1,D1        ; SUBTRACT 1 FROM THE D1 REMOVES
           MOVE.W     D1,(A0)      ; THE LF FROM THE STRING COUNT.
           ; PUT NEW STRING LENGTH IN BUF_POS
;
; TRY TO OPEN A CHANNEL TO THE FILE
           LEA.L      BUF_POS,A0   ; BUF_POS IN A0
           MOVEQ      #1,D3        ; OPEN OLD SHARED FILE
           MOVEQ      #-1,D1       ; JOB ID FOR THIS JOB
           MOVEQ      #1,D0        ; #ID_OPEN IN D0
           TRAP      #2
           TST.L      D0           ; ERROR?
           BEQ.S      GOT_FILE     ; IF NOT, THEN CONTINUE. ELSE:-
;
; IF ERROR, THEN THIS IS USED ( OR BRANCH TO CODE TO KILL JOB)
           BRA        MESS         ; GO BACK TO PROMPT
;
; IF NO ERROR, THEN TRANSFER BYTES FROM FILE TO SCREEN.
.GOT_FILE   MOVE.L     #$20000,A1   ; START ADDRESS OF SCREEN
           MOVE.W     #$FFFF,D3    ; INFINITE TIMEOUT
           MOVE.L     #$8000,D2    ; SCREEN SIZE IN D2
           MOVEQ      #$48,D0      ; FS.LOAD IN D0
           TRAP      #3
           MOVEQ      #2,D0        ; #ID_CLOSE IN D0
           TRAP      #2
;
; *****
; TO BE POSITIONED AFTER THE CODE
; WE NEED 'BUFFER' FOR THE STRING AND 'BUF_POS' FOR THE STRING COUNT
;
.BUF_LEN    EQU      50          ; LENGTH OF INPUT BUFFER
;
.BUF_POS    DC.W      0          ;
;
.BUFFER     EQU      *          ;

```

the bytes are transferred from screen ram to file, using the trap 3 call FS.SAVE, which has #549 in register D0. This also requires the start address to be saved

in register A1, the number of bytes to be saved in D2, with an infinite timeout in D3. This code can be used to save any chunk of ram to a file, and works like the SBYTES command in Basic.

Listing two gives the corresponding Screenload routine. Most of it is very similar to Screensave, except that as it does not have to wait for a screen to be generated, there is no need to have the second Message, or the Priority and Keyrow chunks included. Obviously, it will need a slightly different prompt on the screen, and it will open an old, shared file, not a new exclusive one. And, of course, it will transfer bytes from file to screen ram, and not the other way round. To do this it uses the trap 3 call FS.LOAD, which is identical to FS.SAVE, except that it has #548 in register D0.

This works like the LBYTES command from Basic. We can illustrate the use of these routines with a program which will sleep in the background, waiting to be called to either save a screen to file, or to load a file to screen. It will then wake up and send suitable prompts to the screen, after which it will carry out the appropriate save or load. When this is completed, it will go back to sleep, waiting for the next call.

The program starts in the usual way with Jobstart, and at once it opens a suitable console window for the prompts and file names using Console. It then uses Priority to drop its priority to 1 while it waits to be called. At this stage, the program is looking for three possible key presses, Ctrl-S for save, Ctrl-L for load, or Ctrl-Q for quit.

Different Keyrows

Since Ctrl-S, L, and Q are all in different keyrows, it needs to use Keyrow four times. The first Keyrow is to detect the Ctrl key and ignore all others. When a Ctrl is detected, the other three Keyrows are used to see if S, L or Q is pressed at the same time. If none of these is pressed, it loops back to test for Ctrl again. If Ctrl-Q is detected, the console channel is closed and the job killed.

On detecting Ctrl-S, the program jumps to Priority to bring the priority back to 32 so it can work, and goes on to Screensave. On completing the Screensave routine, the program has to loop back to the first Priority, so it can go back to sleep and wait for another call. So the single line BRA Priority is added to the end of

```

;
;      **** NOTE ****  BUFFER SHOULD COME LAST OF ALL.
;
; *****
Listing 3
; *****
;      'JOBSTART'
;
;      BRA.S      START      ; BRANCH TO START OF CODE
;      DC.L      0          ; (THIS IS STANDARD FORMAT FOR
;      DC.W      $4AFB      ; START OF A JOB)
; ---- ALTER CHARACTER COUNT AND JOB NAME ----
;      DC.W      8          ; CHARACTER COUNT OF JOB NAME
;      DC.B      'SAVELOAD' ; NAME OF JOB
;
; *****
;      'CONSOLE'
;
; OPEN THE CONSOLE CHANNEL
; ---- ALTER LABEL TO .START ----
.START      LEA.L      PBLOCK,A1      ; PBLOCK ADDRESS IN A1
;      MOVE.W      $C6,A2      ; UT_CON VECTOR IN A2
;      JSR          (A2)
;
; SAVE THE CHANNEL ID WHICH UT_CON ROUTINE LEAVES IN A0.
;
;      MOVE.L      A0,-(A7)      ; SAVE CONSOLE ID ON STACK
;
; CLEAR THE WINDOW
;
;      MOVE.W      #$FFFF,D3      ; INFINITE TIMEOUT
;      MOVEQ      #$20,D0      ; #SD_CLEAR IN D0
;      TRAP        #3
;      BRA.S      PRIORITY      ; SKIP BLOCK
;
; DEFINITION BLOCK.
; ---- ALTER WIDTH TO 512 ----
.PBLOCK      DC.B      4          ; GREEN BORDER
;      DC.B      2          ; 2 PIXELS WIDE
;      DC.B      0          ; BLACK PAPER/STRIP
;      DC.B      7          ; WHITE INK
;      DC.W      512         ; WIDTH
;      DC.W      100         ; HEIGHT
;      DC.W      0          ; X POSITION
;      DC.W      0          ; Y POSITION
;
; *****
;      'PRIORITY'
;
; .PRIORITY      MOVEQ      #$B,D0      ; #MT_PRIOR IN D0
;      MOVEQ      #-1,D1      ; OF THIS JOB
;      MOVEQ      #1,D2      ; TO 1
;      TRAP        #1
;
; *****
;      'KEYROW'
;
; - ALTER LABEL TO KEYROW1 AND LOAD ADDRESS TO KEYR1 ----
.KEYROW1      LEA.L      KEYR1,A3      ; KEYROW PARAM TABLE IN A3
;      MOVEQ      #$11,D0      ; #IPCOM IN D0
;      TRAP        #1
;      CMPI.B      #2,D1      ; IS <CTRL> PRESSED?
; ---- ALTER BRANCH ADDRESS IN NEXT TWO LINES ----
;      BEQ.S      KEYROW2      ; IF PRESSED, MOVE ON
;      BRA.S      KEYROW1      ; IF NOT, TEST AGAIN
;
; KEYROW PARAMETER TABLE
; ---- ALTER LABEL TO .KEYR1 ----
.KEYR1      DC.B      9          ; READ KEYROW
;      DC.B      1          ; ONE PARAMETER
;      DC.L      0          ; LOWEST FOUR BITS ONLY
; ---- ALTER NEXT PARAMETER TO 7 ----
;      DC.B      7          ; TEST ROW 7
;      DC.B      2          ; REPLY OF 8 BITS IN D1
;
; *****
;      'KEYROW'
;
; ---- ALTER LABEL TO .KEYROW2 AND LOAD ADDRESS TO KEYR2 ----
.KEYROW2      LEA.L      KEYR2,A3      ; KEYROW PARAM TABLE IN A3
;      MOVEQ      #$11,D0      ; #IPCOM IN D0
;      TRAP        #1
;      CMPI.B      #8,D1      ; IS S PRESSED?
; ---- ALTER BRANCH ADDRESS IN NEXT TWO LINES ----
;      BEQ      SSAVE      ; IF PRESSED, SSAVE
;      BRA.S      KEYROW3      ; IF NOT, TEST AGAIN
;
; KEYROW PARAMETER TABLE
; ---- ALTER LABEL TO .KEYR2 ----
.KEYR2      DC.B      9          ; READ KEYROW
;      DC.B      1          ; ONE PARAMETER
;      DC.L      0          ; LOWEST FOUR BITS ONLY
; ---- ALTER NEXT PARAMETER TO 3 ----
;      DC.B      3          ; TEST ROW 3
;      DC.B      2          ; REPLY OF 8 BITS IN D1
;
; *****

```

Screensave. In a similar way, if the program detects Ctrl-L, it jumps to Priority to bring the priority back to 32 and then goes on to the Screenload routine. As with Screensave, the Screenload routine needs a BRA Priority line added to the end so that it can loop back to wait for another call.

The program ends in the normal way with Close to close the console channel, and Endjob to kill the job. So, putting all that together as list rather than a flowchart, we get:

Jobstart, Console, Priority, Keyrow, Keyrow, Keyrow, Priority, Screensave, Priority, Screenload, Close, Endjob.

Listing three shows all the chunks merged together to make the program. As before, many of the comments have been removed, and extra ones added to show exactly what alterations are necessary to make the chunks fit together. You could, of course, simply type in listing three and assemble the program, but this would defeat the point of the exercise. It will be much more informative to merge the individual chunks into one file, and then make all the necessary alterations.

Tailoring

Having got the program to work, the next job is to improve it and tailor it to your particular needs. Doing this will really show you that you understand the programming.

One obvious improvement might be to add some sound, by getting the program to make a pleasant little sound when it has completed a save or load successfully, and a less pleasant one when it fails to open a file for saving or loading. If you have been following this series, you should have the SOUND routine all ready to merge into the program.

When the program starts, it produces a blank console window, even though you are not ready to use it. This is not a major problem, but may be a bit annoying. Not only that, but the console window stays open the whole time, whether it is in use or not. It would be tidier for the window to be opened when it is needed for a load or save, and then closed again afterwards. This would make the program a bit longer, but would not be difficult to produce. However, if you did this, you would need to be given some sign that the program had loaded properly, so make it give another little sound just before it goes to sleep, just to tell you all is well.

You can, of course, play around with the console window itself

```

;
;                                     'KEYROW'
;
; ---- ALTER LABEL TO .KEYROW3 AND LOAD ADDRESS TO KEYR3 ----
.KEYROW3      LEA.L    KEYR3,A3 ; KEYROW PARAM TABLE IN A3
               MOVEQ   #$11,D0 ; #IPCOM IN D0
               TRAP    #1
               CMPI.B   #1,D1   ; IS L PRESSED?
; ---- ALTER BRANCH ADDRESS IN NEXT TWO LINES ----
               BEQ     SLOAD    ; IF PRESSED, SLOAD
               BRA.S    KEYROW4 ; IF NOT, TEST AGAIN
; KEYROW PARAMETER TABLE
; ---- ALTER LABEL TO .KEYR3 ----
.KEYR3        DC.B     9        ; READ KEYROW
               DC.B     1        ; ONE PARAMETER
               DC.L     0        ; LOWEST FOUR BITS ONLY
; ---- ALTER NEXT PARAMETER TO 4 ----
               DC.B     4        ; TEST ROW 4
               DC.B     2        ; REPLY OF 8 BITS IN D1
;
; *****
;                                     'KEYROW:'
;
; ---- ALTER LABEL TO .KEYROW4 AND BRANCH ADDRESS TO KEYR4 ----
.KEYROW4      LEA.L    KEYR4,A3 ; KEYROW PARAM TABLE IN A3
               MOVEQ   #$11,D0 ; #IPCOM IN D0
               TRAP    #1
               CMPI.B   #8,D1   ; IS 0 PRESSED?
; ---- ALTER BRANCH ADDRESS IN NEXT TWO LINES ----
               BEQ     CLOSE    ; IF PRESSED, QUIT
               BRA.S    KEYROW1 ; IF NOT, TEST AGAIN
; KEYROW PARAMETER TABLE
; ---- ALTER LABEL TO .KEYR4 ----
.KEYR4        DC.B     9        ; READ KEYROW
               DC.B     1        ; ONE PARAMETER
               DC.L     0        ; LOWEST FOUR BITS ONLY
; ---- ALTER NEXT PARAMETER TO 6 ----
               DC.B     6        ; TEST ROW 6
               DC.B     2        ; REPLY OF 8 BITS IN D1
;
; *****
;                                     'PRIORITY'
;
; ---- ALTER LABEL TO .SSAVE ----
.SSAVE        MOVEQ   #$B,D0    ; #MT_PRIOR IN D0
               MOVEQ   #-1,D1    ; OF THIS JOB
               MOVEQ   #1,D2     ; TO 1
               TRAP    #1
;
; *****
;                                     'SCREENSAVE'
;
; PUT PROMPT ONTO SCREEN
;
; ---- ALTER LABEL TO .MESS1 ----
.MESS1        MOVE.L   (A7),A0   ; CHANNEL ID IN A0
; ---- ALTER LOAD ADDRESS TO MESSAGE1 ----
               LEA.L    MESSAGE1,A1 ; BASE ADDRESS IN A1
               MOVE.W   $D0,A2    ; UT_MTEXT VECTOR IN A2
               JSR      (A2)
; ---- ALTER BRANCH ADDRESS TO INSTR1 ----
               BRA.S    INSTR1    ; SKIP MESSAGE
; ---- ALTER LENGTH OF STRING AND STRING BYTES ----
.MESSAGE1     DC.W     44        ; LENGTH OF MESSAGE
               DC.B     'Name of new screen file'
               DC.B     10        ; CR,LF
               DC.B     '(including device): '
;
; GET STRING FROM KEYBOARD
;
; ---- ALTER LABEL TO INSTR1 ----
.INSTR1       MOVEQ   #BUF_LEN,D2 ; LENGTH OF BUFFER IN D2
               MOVE.W   #$FFFF,D3 ; INFINITE TIMEOUT
               LEA.L    BUFFER,A1 ; BASE ADDRESS OF BUFFER IN A1
               MOVEQ   #2,D0      ; #IO_FLINE IN D0
               TRAP    #3
               LEA.L    BUF_POS,A0 ; BUF_POS IN A0
               SUBQ.L   #1,D1      ; SUBTRACT 1 FROM THE D1 REMOVES
                                   ; THE LF FROM THE STRING COUNT.
               MOVE.W   D1,(A0)    ; PUT NEW STRING LENGTH IN BUF_POS
;
; TRY TO OPEN A CHANNEL TO THE FILE
;
               LEA.L    BUF_POS,A0 ; BUF_POS IN A0
               MOVEQ   #2,D3        ; OPEN NEW EXCLUSIVE FILE
               MOVEQ   #-1,D1       ; JOB ID FOR THIS JOB
               MOVEQ   #1,D0        ; #IO_OPEN IN D0
               TRAP    #2
               TST.L    D0          ; ERROR?
; ---- ALTER BRANCH ADDRESS TO CONT1 ----
               BEQ.S    CONT1       ; IF NOT, THEN CONTINUE. ELSE:-
;
; IF ERROR, THEN THIS IS USED
; ---- ALTER BRANCH ADDRESS TO MESS1 ----
               BRA     MESS1        ; GO BACK TO PROMPT
;

```


to your heart's content, by altering its size, shape, position and colours, until it is exactly what you want.

Incorporating

When you are really starting to feel confident, you could start trying to incorporate the routines into other programs. A good candidate for this is the program Ellipses from part 8 of this series. The simplest way of doing this

would be to treat the program Ellipses as a subroutine of Screensave, so that having opened a file to save the screen, the program then jumps to the routines to plot a pattern of ellipses. When a satisfactory pattern is obtained, the program returns to Screensave to save the screen to the file. To make sense of the what follows, you will need to be familiar with the assembler listing for Ellipses.

Clearly, you would not need the whole of the Ellipses program. The Jobstart would be redundant, and you could use the same console window for the prompts and for the ellipses, although this is probably not advisable. Similarly, the Endjob would be redundant. The Close routine in Ellipses to close the console window for drawing ellipses would need to end in an RTS to return to Screensave. The Screensave routine itself would, of course, not need the middle section concerned with changing priority and monitoring the keyboard. Indeed, since sleeping in the background will not feature in the program, then Priority will not be needed. Putting all that together, the list of chunks for merging should look something like this:

Jobstart, Console (for text), Screensave (first part), Console (for ellipses), Cursor, Writemode, Copy, Main (for ellipses routine), Fbyte, Ellipse, Updownfp, Updownint, Updownint, Updownint, Updownint, Close (ellipses console), Screensave (last part), Close (text console), Endjob.

If you try this, you have to be careful in one or two places watching the stack pointer to make sure you put the correct channel ID into register A0. You will also become aware that the programming is becoming just a bit messy because the 'systematic' bit in the title of this series is breaking down. The reason for this is quite simple. As long as our chunks of code remain short and simple they are easy to put together without much change. As soon as our chunks become longer and more complicated, they also become far more specific in application, and when we want to use them in a different way,

```
; IF NO ERROR, THEN THIS CODE IS USED
; ---- ALTER LABEL TO .CONT1 ----
.CONT1      MOVE.L    A0,-(A7)    ; SAVE FILE CHANNEL ID
.MESS2      MOVE.L    4(A7),A0    ; CHANNEL ID IN A0
            LEA.L      MESSAGE2,A1 ; BASE ADDRESS IN A1
            MOVE.W     $D0,A2     ; UT_MTEXT VECTOR IN A2
            JSR        (A2)

; ---- ALTER BRANCH ADDRESS TO PRIOR1 ----
            BRA.S      PRIOR1     ; SKIP MESSAGE
.MESSAGE2    DC.W      30         ; LENGTH OF MESSAGE
            DC.B       'NOW CREATE SCREEN TO BE SAVED'
            DC.B       10         ; CR,LF

;
; REDUCE THE PRIORITY TO 1
; ---- ALTER LABEL TO .PRIOR1 ----
.PRIOR1      MOVEQ     #$B,D0     ; #MT_PRIOR IN D0
            MOVEQ     #-1,D1     ; OF THIS JOB
            MOVEQ     #1,D2     ; TO 1
            TRAP       #1

;
; MONITOR THE KEYBOARD FOR <CTRL><SHIFT>X
; ---- ALTER LABEL TO .KEYROW5 AND LOAD ADDRESS TO .KEYR5 ----
.KEYROW5     LEA.L      KEYR5,A3   ; KEYROW PARAM TABLE IN A3
            MOVEQ     #$11,D0    ; #IPCOM IN D0
            TRAP       #1

; ---- ALTER TO CMPI.B #11,D1 ----
            CMPI.B     #11,D1     ; IS <CTRL><SHIFT>X PRESSED?
; ---- ALTER BRANCH ADDRESS IN NEXT TWO LINES ----
            BEQ.S      PRIOR2     ; IF PRESSED, MOVE ON
            BRA.S      KEYROW5    ; IF NOT, TEST AGAIN

;
; KEYROW PARAMETER TABLE
; ---- ALTER LABEL TO .KEYR5 ----
.KEYR5       DC.B      9         ; READ KEYROW
            DC.B      1         ; ONE PARAMETER
            DC.L      0         ; LOWEST FOUR BITS ONLY

; ---- ALTER NEXT PARAMETER 7 ----
            DC.B      7         ; TEST ROW 7
            DC.B      2         ; REPLY OF 8 BITS IN D1

;
; INCREASE PRIORITY BACK TO 32
.PRIOR2      MOVEQ     #$B,D0     ; #MT_PRIOR IN D0
            MOVEQ     #-1,D1     ; OF THIS JOB
            MOVEQ     #32,D2     ; TO 1
            TRAP       #1

;
; TRANSFER BYTES FROM FILE TO SCREEN
.TRA         MOVE.L     (A7)+,A0   ; FILE CHANNEL ID IN A0
            MOVE.L     #$20000,A1  ; START ADDRESS OF SCREEN
            MOVE.W     #$FFFF,D3   ; INFINITE TIMEOUT
            MOVE.L     #$8000,D2   ; SCREEN SIZE IN D2
            MOVEQ     #$49,D0     ; FS.SAVE IN D0
            TRAP       #3
            MOVEQ     #2,D0       ; #IO_CLOSE IN D0
            TRAP       #2

; ---- ADD NEXT LINE TO LOOP BACK ----
            BRA        PRIORITY

; ---- MOVE BUFFER, ETC TO THE END OF THE PROGRAM ----
; *****
;
; ---- ALTER LABEL TO .SLOAD ----
.SLOAD       MOVEQ     #$B,D0     ; #MT_PRIOR IN D0
            MOVEQ     #-1,D1     ; OF THIS JOB
            MOVEQ     #1,D2     ; TO 1
            TRAP       #1

;
; *****
;
; PUT PROMPT ONTO SCREEN
;
; ---- ALTER LABEL TO .MESS3 ----
.MESS3       MOVE.L     (A7),A0    ; CHANNEL ID IN A0
; ---- ALTER LOAD ADDRESS TO MESSAGE3 ----
            LEA.L      MESSAGE3,A1 ; BASE ADDRESS IN A1
            MOVE.W     $D0,A2     ; UT_MTEXT VECTOR IN A2
            JSR        (A2)

; ---- ALTER BRANCH ADDRESS TO INSTR2 ----
            BRA.S      INSTR2     ; SKIP MESSAGE

; ---- ALTER LABEL TO .MESSAGE3 ----
.MESSAGE3     DC.W      46         ; LENGTH OF MESSAGE
            DC.B       'Name of saved screen file'
            DC.B       10         ; CR,LF
            DC.B       '(including device): '

;
; GET STRING FROM KEYBOARD
;
; ---- ALTER LABEL TO .INSTR2 ----
.INSTR2      MOVEQ     #BUF_LEN,D2 ; LENGTH OF BUFFER IN D2
            MOVE.W     #$FFFF,D3   ; INFINITE TIMEOUT
            LEA.L      BUFFER,A1   ; BASE ADDRESS OF BUFFER IN A1
            TRAP       #3
            LEA.L      BUF_POS,A0  ; BUF_POS IN A0
            SUBQ.L     #1,D1       ; SUBTRACT 1 FROM THE D1 REMOVES
; THE LF FROM THE STRING COUNT.
            MOVE.W     D1,(A0)     ; PUT NEW STRING LENGTH IN BUF_POS
```

it usually means that a slightly different chunk would have suited us better.

Messy Solution

Clearly our problem with Screensave is in linking the input of a file name and opening a file, with actually transferring bytes to the file, all in one chunk. It works quite nicely if we separate the save routine and the screen creation routine into different programs, but becomes messy when we try to put them together in the same program.

The only really systematic way out of this is to go back to the smaller chunks of code and merge them together in the way that best suits your purpose. As I have already mentioned, all these smaller chunks except one have been covered in earlier parts of this series.

The exception is the routine to save the screen bytes to a file. This is the routine SAVE shown in **listing four**. Although we have been using it to save the screen ram to a file, it can be used to save any section of ram to a file, including data, or programs. It requires the channel ID of a new, exclusive channel in A0, the start address to be saved in A1 and the number of bytes to save in D2. It also requires an infinite timeout (-1) in D3.

Screenload does not present us with the same problems. However, it does contain a small routine which has not yet been shown separately, which you may wish to use. This is the Load routine shown in **listing five**. Like Save, it does not have to be used to load a file to screen ram, but can be used to load any file, data, or program, to any part of the ram.

Ram warning

But a word of warning is needed. Qdos needs to know what all parts of ram are used for, because if it doesn't, it is quite likely to suddenly grab some for its own use. Also, if you suddenly load a file into an area of ram that Qdos has allocated for a different purpose, you will almost certainly crash the QL. So if you are not loading the screen ram, you must make sure the ram that is being loaded is properly allocated buffer in a program, and is big enough to take the whole file. Load requires almost exactly the same as Save, except that the channel ID in A0 must be for an old, shared file.

One final point. Listings one, two and three are not supposed to represent any optimal or correct way of doing things. They are designed to show what is possible, and to help understand how certain programming problems may be overcome. As you become more familiar with the various routines in this series, then other, shorter, more elegant, solutions will spring to mind. But the basic building blocks will almost certainly remain the same, and listings four and five are two more of those basic building blocks. Happy coding.

```

;
; TRY TO OPEN A CHANNEL TO THE FILE
;
;
; LEA.L      BUF_POS,A0 ; BUF_POS IN A0
; MOVEQ     #1,D3 ; OPEN OLD SHARED FILE
; MOVEQ     #-1,D1 ; JOB ID FOR THIS JOB
; MOVEQ     #1,D0 ; #ID_OPEN IN D0
; TRAP      #2
; TST.L     D0 ; ERROR?
; BEQ.S     GOT_FILE ; IF NOT, THEN CONTINUE. ELSE:-
;
; IF ERROR, THEN THIS IS USED
; ---- ALTER BRANCH ADDRESS TO MESS3 ----
; BRA        MESS3 ; GO BACK TO PROMPT
;
; IF NO ERROR, THEN THIS CODE IS USED
; TRANSFER BYTES FROM FILE TO SCREEN
;
; GOT_FILE
; MOVE.L     ##20000,A1 ; START ADDRESS OF SCREEN
; MOVE.W     ##FFFF,D3 ; INFINITE TIMEOUT
; MOVE.L     ##8000,D2 ; SCREEN SIZE IN D2
; MOVEQ     ##48,D0 ; FS.LOAD IN D0
; TRAP      #3
; MOVEQ     #2,D0 ; #ID_CLOSE IN D0
; TRAP      #2
;
; ---- ADD NEXT LINE TO LOOP BACK ----
; BRA        PRIORITY
;
; ---- DELETE UNWANTED BUFFER ----
;
; *****
; 'CLOSE'
;
; CLOSE
; MOVE.L     (A7)+,A0 ; CHANNEL ID IN A0
; MOVEQ     #2,D0 ; #ID_CLOSE IN D0
; TRAP      #2
; BRA.S     END_JOB
;
; *****
; 'ENDJOB'
;
; JOB_END
; MOVE.W     #CA,A2 ; UT_ERRO in A2
; JSR        (A2) ;
;
; END_JOB
; MOVEQ     #5,D0 ; #MT_FRJOB IN D0
; MOVEQ     #-1,D1 ; ID OF THIS JOB IN D1
; TRAP      #1
;
;
; *****
; TO BE POSITIONED AFTER THE CODE
; WE NEED 'BUFFER' FOR THE STRING AND 'BUF_POS' FOR THE STRING COUNT
;
; BUF_LEN
; EQU        50 ; LENGTH OF INPUT BUFFER
;
; BUF_POS
; DC.W       0 ;
;
; BUFFER
; EQU        * ;
;
;
; ***** NOTE ***** BUFFER SHOULD COME LAST OF ALL.
;
; *****

```

Listing 4

```

; *****
; 'SAVE'
; *****
; THIS ROUTINE WILL SAVE A FILE FROM MEMORY.
; A1 CONTAINS THE BASE ADDRESS OF THE FILE (MUST BE EVEN)
; D2 CONTAINS THE LENGTH OF THE FILE
;
; SAVE
; MOVE.L     (A7),A0 ; OR 4(A7),A0 OR 8(A7),A0 ETC.
; ; CHANNEL ID IN A0
; MOVE.L     ##20000,A1 ; BASE ADDRESS OF FILE
; MOVE.L     ##8000,D2 ; LENGTH OF FILE
; MOVE.W     ##FFFF,D3 ; INFINITE TIMEOUT
; MOVEQ     ##49,D0 ; FS.SAVE IN D0
; TRAP      #3
;
; *****

```

Listing 5

```

; *****
; 'LOAD'
; *****
; THIS ROUTINE WILL LOAD A FILE TO MEMORY.
; A1 CONTAINS THE BASE ADDRESS OF THE FILE (MUST BE EVEN)
; D2 CONTAINS THE LENGTH OF THE FILE
;
; LOAD
; MOVE.L     (A7),A0 ; OR 4(A7),A0 OR 8(A7),A0 ETC.
; ; CHANNEL ID IN A0
; MOVE.L     ##20000,A1 ; BASE ADDRESS OF FILE
; MOVE.L     ##8000,D2 ; LENGTH OF FILE
; MOVE.W     ##FFFF,D3 ; INFINITE TIMEOUT
; MOVEQ     ##48,D0 ; FS.LOAD IN D0
; TRAP      #3
;
; *****

```

DJC

Dilwyn Jones Computing

41 Bro Emrys, Tal-y-Bont,
Bangor, Gwynedd LL57 3YT U.K.
Tel: Bangor (0248) 354023

GRAPHICS

THE PAINTER V4.04 £25.00
[F 512K] 100% machine code art and graphics program, by PROGS of Belgium. Multiple screens, can be mouse controlled, selection of fonts, patterns, many facilities.

THE CLIPART £12.00
[F 128K] 3 disks full of clipart pictures, for use with most DTP/graphics programs.

QRACAL £20.00
[F 512K] Machine code fractals program. Displays Julia or Mandelbrot sets.

QRACAL SCREENS £ 5.00
[F 128K] A disk full of sample screen pictures generated with the QRactal program. Buy this to see the sample pictures, then ask about a discount when buying QRactal itself!

IMAGE PROCESSOR V2 £15.00
[F 256K] Image enhancement, edge detection, zoom and edit, touch up picture generated by digitisers, scanners or other graphics programs. Works with mode 4 or 8 screens.

UPGRADE OLD IMAGE PROCESSOR £10.00
(Send proof of purchase of old version)

PD2 CLIPART £10.00
[F 128K] Clipart made for PD2 Plus, but available in screen format for use with other DTP/graphics programs if required.

SCREEN SNATCHER £10.00
[F IM 128K] 'Grab' screen displays from within other programs that are not able to save their own displays.

TEXT 'N' GRAPHIX £20.00
[F 256K] **NEW!** Allows you to insert screens, or part of QL screen pictures, into a Quill or plain text file printout. Mix NLQ text and graphics in the same printout! Currently only for use with 9 pin Epson compatible dot matrix printers - ask about a version for 24 pin printers due soon.

TRANS24 £10.00
[F IM 128K] Translates 9 pin graphics printouts into data for a 24 pin printer.

TEXT

BIBLE TEXT DISKS, EDITOR FORMAT £20.00
BIBLE TEXT DISKS, QUILL " _DOC" £20.00
[F 256K] Text of the King James Bible on Disk. Please state whether you require Editor (plain text) or Quill " _doc" format.

SPELLBOUND £30.00
[F IM 384K] A spelling checker which can check spelling as you type! 30,000 word dictionary

SPELLBOUND SPECIAL EDITION £50.00
(F 512K) Enhanced version of Spellbound with 50,000 word dictionary & many new features.

UPGRADE TO SPELLBOUND S.E. £30.00

QUICK POSTERS £10.00
[F 2M 128K] Text poster maker, for use with Star NL, XB and LC printers.

ROB ROY BARGAIN PACK £10.00
(F 3M 128K) Reviewed in QL World August 1991

DATA BASES

ADDRESS BOOK & LABEL PRINTER £15.00
[F 2M R 384K] Store names and addresses and print them out on a variety of label sizes, or print a telephone list, etc.

DATA DESIGN £50.00
[F 512K] Superb, fast 100% machine code database by PROGS of Belgium. Programmable via BASIC and machine code interfaces - write your own database applications in BASIC using the DATA Design commands to help you! Runs in pointer environment (supplied)

FLASHBACK £25.00
[F IM 256K] Fast machine code database which is also very easy to use.

FLASHBACK SPECIAL EDITION £40.00
[F 2M 512K] Enhanced version of Flashback, new commands, report generator, etc

UPGRADE TO FLASHBACK S.E. £20.00
(return Flashback master disk)

QL GENEALOGIST, STANDARD VERSION £19.50
[F 2M 384K] Family trees and family history database, one of our best selling programs.

QL GENEALOGIST SECOND EDITION £30.00
[F 384K] Enhanced version - for details see our advert in QL World December 1991.

UPGRADE TO SECOND EDITION £12.00
(return Genealogist master disk)

BUDGET 128K QL GENEALOGIST £12.00
[F IM 128K] Cut down version for unexpanded machines.

SUPER DISK INDEXER £12.00
[F IM 384K] Catalogue your disk collection as a database of files. Store it, print it, search & find files, make your life easier.

DBEASY £15.00
[F 512K] **NEW!** A database front end for Archive, plus a suite of programs called Chaos Busters. Software from EMSOFT, U.S.A.

DBPROGS £15.00
[F 512K] **NEW!** A collection of Archive utilities and text files to help you learn to program Archive. Software from EMSOFT, U.S.A.

DTP

PAGE DESIGNER 2 PLUS £40.00
[F 512K] QL Desktop Publishing program, mix text and graphics, make printed pages or posters, use clipart, use graphics from other programs etc. Prints on most printers. Ask for details! (please note: this program has been delayed, apologies to those waiting - please check if available before ordering)

UPGRADE PD2 TO PD2 PLUS £20.00
(Send proof of purchase of old PD2)

QL HARDWARE

MINI PROCESS CONTROLLER £59.95
Relay switched outputs, controlled via QQL serial port.

SOFTWARE TOOLKIT FOR MPC ABOVE £9.95
[F 128K] Control the MPC from BASIC programs more easily with these extensions.

NETWORK PROVER £ 3.50
Plugs between network lead and computer to give a visual indication of transmission.

POSTAGE RATES - SEE BELOW

OTHER QL SOFTWARE

PRINTERMASTER £20.00
[F IM 128K] **NEW!** Tame your printer! Popup utility with menu selection of control codes to send to a printer. Want to change fonts, set paper length, margins, set up for listings or wordprocessing etc. without having to refer to the manual and struggle with control codes. Let Printermaster take the strain out of living with your printer.

HOME BUDGET £20.00
[F IM 128K] Domestic bills and accounts program, plus a UK Income Tax Calculator.

REMIND-ME £12.00
[F IM 128K] Dates and events reminder program - remember about birthdays, licence renewals, etc. Quick and very easy to use.

REMIND-ME PLUS £20.00
[F IM 128K] **NEW!** Enhanced version with ability to schedule twice as many dates, longer event descriptions, etc.

UPGRADE REMIND-ME TO REMIND-ME PLUS £10.00
Return master disk with order for upgrade

SCREEN ECONOMISER £10.00
[F IM 128K] Turns off the QL display after a set number of minutes to protect the screen.

SLOWGOLD £ 5.00
[F 128K] Slowdown routine and control panel for software which runs too fast on Gold Card or indeed any QL system.

TASKMASTER £25.00
[F IM 384K] Task switching utility. Enables you to conveniently switch between several programs in memory. Calculator, notepad and file handling utility included.

SUPER DISK LABELLER £10.00
[F 256K] Print neat labels for your floppy disks listing the filenames on the disk in columns in small print.

THE CAT £ 5.00
[F IM 128K] List files on a disk or cartridge to the screen, or print on paper, in columns (optionally sorted in this latest version). Useful and convenient utility.

★ SEE ALSO THE OTHER HALF OF THIS ADVERT ON THE PRECEDING PAGE ★



SOFTWARE POSTAGE: Software sent post-free to UK addresses. Abroad add £1.00 per program for postage and packing.
SUPPLIES POSTAGE: For disks, boxes and stands, add £2.50 for postage to UK addresses, or 10% of order value (minimum postage £3.50) for airmail postage where possible.

PAYMENT: We can accept payment by cash, cheque (in UK Pounds Sterling only, drawn on UK branch of bank or building society), Postal Order, International Postal/Money order, or Eurocheque. We can also accept payment by these credit cards: VISA, ACCESS, MASTERCARD, EUROCARD and also Barclays CONNECT card. Please state your card type, number, expiry date and your address (goods paid for by credit card can only be sent to cardholder's address). Remember to sign your order. Goods remain property of DJC until paid for in full. Our telephone number above has an answering machine for when we are unable to answer in person - don't be shy!

